数据之美

Beautiful Data

揭秘优雅的数据解决方案背后的故事

Toby Segaran & Jeff Hammerbacher 编

祝洪凯 李妹芳 段炼 译

O'REILLY®

机械工业出版社
China Machine Press



O'Reilly精品图书系列

数据之美

Beautiful Data:The Stories Behind Elegant Data Solutions

[美]Toby Segaran

Jeff Hammerbacher　编

祝洪凯　李妹芳　段炼　译

ISBN：978-7-111-31512-4

本书纸版由机械工业出版社于2010年出版，电子版由华章分社（北京华章图文信息有限公司）全球范围内制作与发行。

客服热线：+ 86-10-68995265

客服信箱：service@bbbvip.com

官方网址：www.bbbvip.com

新浪微博 @言商书局BBBVIP

腾讯微博 @bbb-vip

目　录

O'Reilly Media,Inc.介绍

为了满足读者对网络和软件技术知识的迫切需求，世界著名计算机图书出版机构O'Reilly Media,Inc.授权机械工业出版社，翻译出版一批该公司久负盛名的英文经典技术专著。

O'Reilly Media,Inc.是世界上在Unix、X、Internet和其他开放系统图书领域具有领导地位的出版公司，同时也是联机出版的先锋。

从最畅销的The Whole Internet User,s Guide＆Catalog（被纽约公共图书馆评为20世纪最重要的50本书之一）到GNN（最早的Internet门户和商业网站），再到WebSite（第一个桌面PC的Web服务器软件），O'Reilly Media,Inc.一直处于Internet发展的最前沿。

许多书店的反馈表明，O'Reilly Media,Inc.是最稳定的计算机图书出版商——每一本书都一版再版。与大多数计算机图书出版商相比，O'Reilly Media,Inc.具有深厚的计算机专业背景，这使得O'Reilly Media,Inc.形成了一个非常不同于其他出版商的出版方针。O'Reilly Media,Inc.所有的编辑人员以前都是程序员，或者是顶尖级的技术专家。O'Reilly Media,Inc.还有许多固定的作者群体——他们本身是相关领域的技术专家、咨询专家，而现在编写著作，O'Reilly Media,Inc.依靠他们及时地推出图书。因为O'Reilly Media,Inc.紧密地与计算机业界联系着，所以O'Reilly Media,Inc.知道市场上真正需要什么图书。

译者序

我一直对数据挖掘很感兴趣，尤其是通过对海量、抽象甚至枯燥的数据进行挖掘分析后，利用数据可视化工具展现出来的那种绚丽多彩、富含意蕴的数据之美更是令我痴迷、叹为观止。本书涉及领域很广，各领域的精英们向我们娓娓道来相关领域的数据信息系统的架构的设计，包括Yahoo！的云存储架构、Deep Web数据抓取、Facebook的信息平台、自然语言处理、“凤凰号”火星探测器的图像数据处理、探索数据生命的DNA漫谈，甚至是Radiohead视频的制作、旧金山的次贷危机等。

阅读完本书之后，我自己的一个很大的收获是对于自己比较了解的领域，如云存储、Deep Web、NLP等有了进一步的理解和实践指导，而对于那些完全不熟悉的领域，如探索数据生命、火星探测器、制作Radiohead视频等则更是开阔了视野，不但对数据有了新的认识，而且激发了思考问题的一些新的思维方式。

这本书令我很感怀的另一方面是，我发现这些“数据科学家”在兢兢业业构建平台处理数据的过程中，虽然遇到了很多困难和挑战，但是却依然如此坚持、执着地探索数据之美。在翻译本书过程中，这种激情不仅激励着我完成这本书的翻译，同时也激励着我在生活、工作中要有毅力和恒心。而纵观我身边的阿里巴巴云计算的同事们——这些“阿里数据科学家”们，也无一不是那种永远充满着激情致力于我们的“飞天”梦想！

这是我翻译的第一本书，很感激机械工业出版社华章公司编辑陈冀康先生慷慨地引我入门，并且对因为我前段时期项目开发非常紧张而导致翻译进度几乎停滞的宽容和理解表示深深感激。感谢所有其他为本书付出努力的人们。

由于时间和精力有限，本书的疏漏、错误之处在所难免，还望各位读者不吝批评指正。

李妹芳

2010年6月26日

前言

当我们第一次接触为《代码之美》编写“续集”的想法时，这次是关于数据也就是这本书，我们觉得这个想法令人兴奋且很有挑战性。现在收集、可视化和处理数据涉及每个专业领域和日常生活的诸多方面，一个大数据集在范围上将是难以想象的广泛。因此，我们联系了一组相当多样化的群体，这些人的工作让我们钦佩。当他们中的大多数都同意撰稿时，我们感到异常兴奋。

这本书就是我们努力的结果，我们希望它能够展示数据处理工作可以多么的广泛（和美丽）。在本书中，你将了解从和政府协作到和火星登陆器一起工作的各个方面；你将了解如何使用统计程序、制作可视化应用、混合Radiohead视频；你将看到地图、DNA和一些我们真正只能称之为“数据哲学”的内容。

本书的版权收益贡献给知识共享组织(Ceative Commons)和阳光基金会(te Sunlight Foundation)，它们致力于通过解放数据使世界变得更美好。我们希望你将会考虑你和数据亲身“邂逅”的经历如何塑造了世界。

本书的组织方式

本书的章节贯彻一条较为松散的曲线：从数据收集到数据存储、组织、检索、可视化及最后的数据分析。

第1章：在数据中观察生活。作者Nathan Yau着眼于在新兴的个人数据收集领域的两个项目背后的动机和挑战。

第2章：美丽的人们：设计数据收集方法时牢记用户。Jonathan Follett和Matthew Holm讨论了在Web上向人们收集数据时，信任、说服和测试的重要性。

第3章：火星上的嵌入式图像数据处理。J.M.Hughes分析了设计在太空旅行下能够正常工作的数据处理系统所面临的挑战。

第4章：PNUTShell中的云存储设计。Brian F.Cooper、Raghu Ramakrishnan和Utkarsh Srivastava描述了雅虎所设计的软件系统，该系统将其全球分布式数据中心转换为支持现代Web应用的通用存储平台。

第5章：信息平台和数据科学家的兴起。Jeff Hammerbacher以Facebook的数据团队的历史演化作为特例，追溯了信息处理工具以及驱动这些工具的人们的演化。

第6章：照片档案的地理之美。Jason Dykes和Jo Wood吸引人们注意一个志愿者组织收集的彩色可视化空间数据的普及性及其力量。

第7章：数据发现数据。Jeff Jonas和Lisa Sokol阐述了思考数据的新方式，为了完全管理这些数据，很多人需要采用这种方式。

第8章：实时的可移动数据。Jud Valeski深入分析了Web上实时的分布式社会和定位数据当前存在的局限，讨论了解决该问题的一个可能方案。

第9章：探寻Deep Web。Alon Halevy和Jayant Madhavan描述了G公司开发的用于搜索当前“受困”于Web表单之后的数据的工具。

第10章：构建Radiohead的“House of Cards”。Aaron Koblin和Valdean

Klump讲述了一个涉及激光、编程和“骑在巴士背上”的惊险故事，故事以一个获奖音乐视频结束。

第11章：都市数据可视化。Michal Migurski详细描述了释放和美化一些我们身边的最重要的数据的过程。

第12章：Sense.us的设计。Jeffrey Heer重塑了作为社会空间的数据可视化，并使用这种新视角来探索历时150年的美国人口普查数据。

第13章：数据所做不到的。Coco Krumme关注于证明人们在很多方面误解和误用数据的实验性工作。

第14章：自然语言语料库数据。Peter Norvig通过从Web上获取的1兆规模的自然语言词汇语料数据，带领读者走进一些令人回味的实践。

第15章：数据中的生命：DNA漫谈。Matt Wood和Ben Blackburne描述了数据之美，即DNA和创造、捕捉和处理数据需要的大量基础设施。

第16章：美化真实世界中的数据。Jean-Claude Bradley、Rajarshi Guha、Andrew Lang、Pierre Lindenbaum、Cameron Neylon、Antony Williams和Egon Willighagen展示了“众包”(cowdsourcing)和高度透明的结合如何提高了药物发现的研究。

第17章：数据浅析：探索形形色色的社会定型。Brendan O'Connor和Lukas Biewald展示了当让人们匿名对其他人的图片进行打分时所表现出来的关联和模式。

第18章：旧金山海湾之殇：次贷危机的影响。Hadley Wickham、Deborah F.Swayne和David Poole通过使用开源软件和公共数据资源，带

领读者走进对近年来旧金山海湾地区的住房危机的详尽研究。

第19章：美丽的政治数据。Andrew Gelman、Jonathan P.Kastellec和Yair Ghitza展示了统计和数据可视化工具是如何帮助我们加深对社会进行组织的政治进程的理解。

第20章：连接数据。Toby Segaran探索了对Web上可获取的大量的数据集进行连接的挑战性和可能性。

本书使用的体例

本书遵循以下字体体例：

斜体(Ialic)

表示新的术语、URL、Email地址、文件名和文件扩展名。

等宽字体(Cnstant width)

用于程序清单以及段落中的程序单元如变量或函数名称、数据库、数据类型、环境变量、声明和关键字。

等宽粗体字(Cnstant width bold)

显示命令或者其他由用户输入的文本。

等宽斜体字(Cnstant width italic)

表示必须根据用户提供的值或者由上下文决定的值进行替代的文本。

使用本书的样例代码

本书是为了帮助你完成工作。通常来说，你可以在你的程序和文档中使用本书的代码。除非你使用了本书的大量代码，否则你无需联系我

们获取许可。例如，写一个程序用到本书的几段代码不需要获得许可，销售和分发O'Reilly丛书的代码需要获得许可；引用本书的样例代码来解决一个问题不需要获得许可，使用本书的大量代码到你的产品文档中需要获得许可。

我们不要求你（引用本书时）给出出处，但是如果你这么做，我们对此表示感谢。出处通常包含标题、作者、出版社和ISBN。例如："Beautiful Data,edited by Toby Segaran and Jeff Hammerbacher.Copyright 2009 O'Reilly Media,Inc.，978-0-596-15711-1."。

如果你觉得你对本书样例代码的使用超出了这里给出的许可范围，请与我们联系：permissions@oreilly.com。

联系方式

如果您对本书有任何意见和问题，请联系出版社：

美国：

O'Reilly Media,Inc.

1005 Gravenstein Highway North

Sebastopol,CA 95472

中国：

北京市西城区西直门南大街2号成铭大厦C座807室（100035）

奥莱利技术咨询（北京）有限公司

O'Reilly的每一本书都有专属网站，你可以在那找到关于本书的相关

信息，包括勘误列表、示例代码以及其他的信息。本书的网站地址是：

http://www. oreilly.com/catalog/9780596157111/

对于本书的评论和技术性的问题，请发送电子邮件到：

bookquestions@oreilly. com

关于本书的更多信息、会议、资料中心和网站，请访问以下网站：

http://www. oreilly.com

http://www. oreilly.com.cn

第1章　在数据中观察生活

NathanYau

在不远的过去，Web就是关于共享、广播和分发。但是潮流在变：Web已经走进了个人的世界。每个月，新的应用如雨后春笋般诞生，这些应用可以追踪、监测和分析人们的行为习惯，使他们更好地了解自己和周围的世界。人们可以对饮食习惯、运动、上网时间、性生活、每月的自行车旅行、睡觉、情绪和资产进行在线跟踪。如果你对自己生活中的某一部分感兴趣，很可能存在某种应用可以对它进行跟踪。

当然，个人数据收集不是什么新鲜事。在20世纪30年代，英国的社会研究小组Mass Observation，收集了关于日常生活的各个方面的数据——如胡须、眉毛、驾驶员的叫喊和手势，以及人们在战争纪念碑前的行为，这些都是为了对英国有更好的理解。然而，数据收集方法从1930年开始已经有了提高；现在不再仅仅是一支笔和便条纸，或者是一个手工的计数器。数据可以通过手机和手持电脑自动收集，因此每天都可以持续地上传数据和信息流到服务器、数据库和所谓的数据仓库。

随着数据收集技术的进步，数据流也发展为比Mass Observation[1]参与者报告的计算计数要强大得多的技术。数据可以实时修改，因此，人们期望获取最新的信息。

但是，只是简单地为人们提供数以GB计的数据是不够的。并非所有人都是统计学家或者计算机科学家，而且也不是每个人都愿意对大数据

集的数据进行筛选过滤。这是我们在个人数据收集上经常面临的问题。

虽然收集和返回的数据类型可能已经随着时间变化，但是个人的需求是不变的。也就是说，那些收集关于自己和他们周围的数据的个人，他们还是会收集这些数据，以获得对流动的数据的信息有更好的理解。绝大多数时候，我们不是追求数据本身；我们感兴趣的是数据的真正含义。这是个微小的区别，却是非常重要的一点。这个需求要求系统能够处理个人数据流，高效准确地处理这些数据，把这些信息通过易于理解且有用的方式分发给非专业人员。我们想要的远远不只是一个电子表格的数据，我们想要的是隐含在这些数据中的故事。

构建这样的系统要求同时在分析和审美上作出严谨的设计思考。这在我们实现PEIR(Personal Environmental Impact Report，个人环境影响报告）中是非常重要的。PEIR允许人们在微观层面观察自己如何影响环境以及环境如何影响自己；your.flowingdata(YFD)是一个正在开发的项目，允许用户通过Twitter来收集自己的数据，后者是一个微博客服务。

在PEIR项目中，我是其前端开发工程师，主要工作是负责用户界面(U)和数据可视化。对于YFD，我是唯一参与者，因此我的职责可能有些不同，但我还是集中于研究可视化方面。虽然PEIR和YFD在数据类型、收集和处理上区别甚大，但是它们的目标是相似的。PEIR和YFD都是为个人提供信息；但为个人提供信息又都不是它们的最终目标。相反地，它们都是为了激发人们的兴趣，了解即自己每天做出的抉择如何为生活带来重大影响，然后开始探讨个人数据。在对PEIR和YFD有了简单

的背景知识后，我将带着这些想法，讨论个人数据收集、存储和分析。然后我将深入探讨PEIR和YFD数据可视化背后的设计过程，通常这些统称为个人数据可视化。最后，我们想向人们显示他们自己的个人数据之美。

个人环境影响报告(PIR)

PEIR是由加州大学洛杉矶分校的嵌入式网络传感中心(Cnter for Embedded Networked Sensing)开发的，或者更准确地说，是由城市传感小组(Uban Sensing group)开发。我们重点利用日常的移动技术（如手机）来收集关于周围和自己的数据，因此人们可以对如何与身边的事物进行交互有更好的理解。例如，DietSense是一个在线服务，它允许人们自我监测饮食选择以及进一步向饮食专家咨询；Family Dynamics帮助家庭和生活教练记录一个家庭每日交互的关键特征，如户外驻扎和家庭聚餐；Walkability帮助居民和行人，提倡通过观察发表他们对于附近的步行适宜性和与公共交通的联系的看法。[2]所有这些项目使得人们可以只通过手机参与到社区活动中。我们利用手机内置的传感器，如摄像头、GPS和加速度计来收集数据，通过这些设备来提供信息。

PEIR采用了类似的原则。例如某个用户下载一个名为Campaignr的很小的软件到他的手机，该软件可以在后台运行。随着该用户每日的活动——比如在田径道慢跑、开车上下班、去杂货店买点东西，手机每2分钟就会上传GPS数据到PEIR的中央服务器，包括纬度、经度、高度、速率和时间。我们利用该数据来预估个人对环境的影响和排放。不需要环境污染传感器；相反地，我们使用在很多手机上已经具备的功能——GPS，然后传递这些包含“环境”信息的数据（如天气）来构建环境模型。最后，我们对环境影响和排放数据进行数据可视化。这个阶段的挑

战是向人们传达数据中绝大部分人都不熟悉的含义。每周释放1000千克的碳意味着什么？1000千克是很多还是很少？我们需要谨记我们的用户和目标，因为用户和目标驱动着系统设计，从可视化直到数据收集和存储。

[1]Mass Observation是一个英国社会研究组织。它通过500名未经过训练的志愿者的观察来记录英国人的日常生活。请参考http：//en.wikipedia.org/wiki/Mass_Observation。

[2]CENS城市感知，具体参见http：//urban.cens.ucla.edu/。

your.flowingdata(YFD)

PEIR使用的是运行在后台的定制软件，YFD则要求用户主动通过Twitter输入数据。Twitter是一个微博客服务，它提出了一个简单的问题：您现在在做什么？人们可以张贴，或者更确切地说，可以通过桌面应用、邮件、即时通信告诉别人他们正在做的事；而且最重要的是（就YFD而言）SMS，它意味着人们可以通过手机和别人交流。

YFD利用了Twitter的普遍性，因此人们可以在任何地方通过Twitter发送SMS消息来交流个人数据。用户目前可以跟踪饮食习惯、体重、睡眠、情绪，以及简单地通过发布特定形式的消息来跟踪他们什么时候上洗手间。像PEIR一样，YFD向用户表明的是：这些小事可以对我们的生活方式产生深远的影响。在设计过程中，我们依然考虑用户体验。什么将会激励用户频繁地手工输入数据？我们如何使得数据收集对于用户来说尽可能地简单？一旦数据被记录下来，我们应该如何和用户交流？最后，我将从数据收集开始谈起。

个人数据收集

个人数据收集和科学数据收集在一定程度上有所区别。个人数据收集通常较为不正式，而且不是发生在具备控制条件下的实验室里。人们收集现实世界中的数据，而在现实世界中，存在干扰、网络连接故障或者访问受限的计算机。用户不一定是数据专家，所以如果出了问题（这通常是不可避免的），他们可能不知道如何调整来解决问题。

因此，从用户角度，我们需要使得数据收集变得尽可能简单。它应该是无干扰的、直观的且易于访问的，这样数据收集才更有可能成为日常生活的一部分。

使数据收集成为日常生活

这是我选择Twitter作为YFD数据代理的主要原因，它可以通过电话或者计算机连到数据库。Twitter允许用户通过一些不同的渠道发布Twitter消息(teet)。用户可以通过移动电话发布Twitter消息，由于这一点，用户可以在任何地方通过电话发送SMS消息记录数据，这意味着人们可以随时记录实际情况，而不需要等到他们可以访问计算机时才开始记录。如果需要等待，人们很可能会忘记。因此，易于访问获取是很关键的。

人们可以通过E-mail而不是Twitter来完成一些相似的事情，因为很多手机允许人们给E-mail发送SMS，而这实际上正是YFD的最初实现方式。然而，我们回到数据收集作为日常生活的一个基本部分。成百上千

万的人们已经在频繁地使用Twitter，由于这一点，其挑战性已经被部分削弱了。人们确实也在很频繁地使用E-mail，而且比起Twitter，人们也有可能更习惯于使用E-mail。但是这二者在本质上还是很有区别。在Twitter上，人们发布自己正在做的事，每天更新好多次信息。Twitter最初被创建也仅仅是因为这个原因。某个人可能正在吃三明治、出去散心或者在看电影。成千上万的人每天发布这类Twitter消息。而另一方面，E-mail本身就意味着其主要是一些更有价值的信息。绝大多数人不会给朋友发E-mail，告诉朋友他们正在看一个电视节目——尤其不会每天或者每个小时告诉朋友们这些事情。

通过利用Twitter得到的这种发布频率，我们期望它能够用于数据收集。我尝试着使得在YFD上记录数据和在Twitter上的感觉一样。举个例子，如果有个人吃意大利香肠的三明治，他发送了一条消息："吃了意大利香肠三明治"(ae salami sandwich)。通过这种方式，数据收集就变成了交谈方式。用户不需要学习像SQL一样新的语言。相反地，他们只需要记住关键字及其对应值。在前一个例子中，关键字是"ate"，值是"salami sandwich"。为了跟踪睡眠，用户简单地发送一个关键字：睡觉时是"goodnight"，醒来时是"gmorning"。

在某些方面，用PEIR频繁发布信息的挑战要比在YFD上小一些。因为PEIR是在后台自动收集数据，用户只需要通过点击几下按钮，就可以启动其手机上的软件。开发那种软件本身存在一定的困难，但那就是另一回事了。

异步数据收集

对于PEIR和YFD，我们发现异步数据收集实际上是必要的。当人们遇到一些感兴趣的事情发生后，他们会希望输入和上传数据。在YFD，人们希望能够给他们的Twitter消息增加时间戳，而PEIR用户想要手工上传GPS数据。

正如之前所述，创建YFD的初始想法是人们在遇到一些事情时，会希望输入数据来记录它们。这是使用Twitter的好处和目的。然而，很多人没有通过手机使用Twitter，因此他们需要等到可以用电脑时才登录Twitter。即使是那些确实给Twitter发送SMS消息的用户通常也会忘记记录数据；有些人只是想在每天结束前输入所有的数据。

不用说，YFD现在支持时间戳。数据入口的语义尽可能地贴近交谈仍然很重要。为了实现这个目标，用户可以给他们的Twitter消息添加时间戳，比如“早上6点吃烤鸡和土豆”(ae roast chicken and potatoes at 6：00pm)或者“晚上11点晚安”(godnight at 23：00）。时间戳语义上只是简单地在Twitter消息的最后添加一个“at hh:mm”。我还发现同时支持标准格式和军事格式的时间戳是有用的。最后，当用户输入一个时间戳，YFD将会记录该时间的最近出现频度，因此在先前的“goodnight”例子中，YFD会输入昨天晚上的时间点。

PEIR的初始设计也只是为了“当时”(i the moment)的数据收集。正如前述，Campaignr运行在用户的手机上，周期性把GPS数据（每20分钟上传一次）上传到中心服务器。这对一个每天运行PEIR的用户来说，他

本身只需要很少的付出，就可以收集成千上万的数据点。一旦在手机上安装PEIR应用，用户只是简单地按动几下按钮就可以启动应用。然而，几乎从一开始，我们就发现不能依赖100%可达的网络连接，因为几乎总有地区是服务无法覆盖的。最简单虽然也是最幼稚的做法是只有当手机可以上网时才收集和上传数据。这种做法的代价是我们可能会失去大量的数据。相反地，我们通过缓存把数据存储在手机的本地内存，直到手机可以重新连上网络。我们还提供了另一个方案来收集数据，而不需要同步上传这些数据。

这种做法的一个不足之处在于，期望人们在事件发生时收集数据是不合理的。人们会忘记收集数据或者在当时不方便收集数据。在以上任何情况下，提供用户在后期也能够输入数据的功能是很重要的，这一点又反过来影响了数据流的下一步设计。

数据存储

对于YFD和PEIR，很重要的一点是应该记住：一旦数据被收集了，我们应该如何处理这些数据。通常情况下，数据库机制和数据库模式都是在一念之间设计的，研究人员在处理过程中会渐渐后悔，其原因或者是因为他们当时的决策使现在的数据处理变得很困难，或者是因为数据库不具有扩展性。YFD在这方面的选择不是特别困难。我们在其他项目中使用MySQL，而YFD通常只涉及较简单的插入和选择语句，因此很容易搭建系统。此外，数据是人工输入的——而不是像PEIR那样连续上传，因此数据库表的大小在开发早期不是问题。主要的考虑是当增加新的数据字段时，我希望能够扩展数据库模式，因此在创建数据库模式时考虑到这一点。

另一方面，PEIR需要更细心的数据库开发。我们每几分钟就执行成千上万的基于地理的计算，因此我们采用了PostGIS，为PostgreSQL数据库增加地理对象支持。虽然MySQL提供了GIS和空间扩展，我们认为从PEIR的需求考虑，把PostGIS和PostgreSQL结合在一起是更健壮的方案。

以上的描述可能过于简化了我们的数据库设计过程，这里我需要谈一点背景。我们的团队是由10个左右的研究生组成，每个人都有自己的研究方向。正如你所料，每个人也都致力于PEIR的不同模块的工作。这种分工方式极大地影响了我们的工作方式。PEIR的数据最初是非常分散

的。我们没有使用统一的数据库模式；在需要多个数据库时，就创建多个数据库，而且并没有遵循任何特定的设计模式。无论谁在PEIR的早中期加入PEIR项目，他都会对去哪里查找数据、查找什么数据以及应该和谁联系感到困惑不已。我这么说是因为我自己正是在这个时候加入PEIR。为了减轻数据分散带来的问题，我们最终冻结了所有的开发工作。一个在PEIR的各个模块都做过开发的同事，很灵活地把每个人的代码和数据库表结合起来。本次代码和模式的凝合（c onsolidation)在用户体验开发之前完成是非常必要的，而且这一点在后期中越来越明显。现在回想起来，在早期对数据存储方面付出更多的努力来进行精心设计是很值得的，而这一点也正是研究生研究的本质。

协调和代码凝合对于YFD不是问题，因为YFD只有我一个开发人员。我可以很容易改变数据库模式、用户接口和数据收集机制。我采用了Django，一个基于Python语言的Web框架，它采用了MVC(Model-View-Control，模型-视图-控制器）模式，支持敏捷高效开发。但是，我确实需要每件事都自己完成。因为我们团队的每个小组成员在统计、计算机科学、工程、GIS和环境科学上的多样性，PEIR可以完成更多的功能——最显著的是在数据处理领域，这在下一节将会介绍。因此，和一个庞大的团队一起开发，必然有其一定的优势和不足。

数据处理

数据处理是个人数据收集系统最重要、最基础的部分，这部分几乎都不为用户所见，而且用户也不感兴趣。用户通常是对数据处理的结果更感兴趣。这就是YFD面临的情况。相反地，PEIR用户可以通过观察他们的数据是如何处理的而从中受益，而且这反过来也影响他们诠释数据表示对环境带来的影响和排放的方式。

PEIR的分析组件包含一系列服务端的处理步骤，这包括GPS数据到对影响和排放的预估。更确切地说，我们可以把PEIR的处理过程分解为四个独立的阶段[1]。

1.跟踪纠正和标注：只要可能，我们通过预估技术，如利用道路网进行地图匹配和构建宗地数据(prcel data)来对易于引起错误、抽样定位的地理位置踪迹进行纠正和标注。因为这些纠正和标注是估计值，它们确实包含一定的不确定性。

2.活动和地理位置分类：通过使用Web服务，纠正和标注的数据会自动分类为移动的或者静止的，从而可以为某人在某天的模型输出提供一级完善。其数据也基于停滞时间，划分为不同的旅途行程。

3.上下文预估：把正确和分类的地理位置数据作为基于Web的信息资源的输入，这些信息资源包括天气、路面状况和驾驶员行为的汇总。

4.排放和影响计算：最后，把细粒度、分类和衍生的数据作为地理空间数据集合和微观环境模型的输入，这些输入反过来又可以用于提供

个人的个性化估计值。

虽然PEIR的重点还是研究这四个步骤的处理过程的结果，但是我们最终发现用户想要对如何估计影响和排放这些方面有更多的了解。因此，我们提供每块数据处理过程的细节，比如在高速公路上花费的时间比、用户旅游地方周边的天气情况。我们还包含了对每个提供的尺度的详细解释。在这种情况下，预估过程的透明性允许用户了解他们的行为对环境是如何产生影响和排放的，而不仅仅是了解他们对周边地区造成的污染有多么大或多么小。当然，也存在如信息过载的情况，因此我们对于应该展示多少信息量很慎重。这些方面将在下一节做更多的探讨。

[1]PEIR,http：//peir.cens.ucla.edu

数据可视化

一旦数据被收集、上传和处理，用户需要能够访问、评估和浏览他们的数据。YFD和PEIR的主要设计目标是使得个人数据对于非专业人员也可以理解。数据必须通过可关联的方式来展示；它需要具有人性化。通常情况下，我们会过于陷入统计图形和图表之中。这些图形和图表非常有用，但是同时我们也希望用户能够参与进来，这样他们可以因为感兴趣而留下来，继续收集数据，不断地回到站点来衡量他们所跟踪的任何事情的进展。用户必须理解的一点是：数据是关于他们自己的信息，而且反映的是他们在日常生活中做出的选择。

我想把数据可视化视为一个故事。故事的主角是用户。我们可以采取两种方式来讲述这个故事。一个关于图形和图表的故事可能读起来很像一本教科书，但是，一个包含场景、关系、交互、模式和解释的故事读起来像是一部小说。这并不是说其中一个比另一个更好。有很多有趣的教科书，同时也有很多（如果不是更多）庸俗的小说。当我们对个人数据作可视化分析时，想要的是介于教科书和小说之间的“故事”。我们想要展示事实，但我们也想提供场景，比如什么人、什么时间、什么地点以及结果数字的原因是什么。我们追求的是情感。数据通常是枯燥无味的，但只有当我们以枯燥的方式来展现时它们才会如此。

PEIR

对于PEIR，我们面临的挑战是如何展示科学数据——“碳”影响、高

密度的颗粒物排放以及对敏感地区（如医院和学校）造成的影响。影响和排放并没有成为人们每天谈资的一部分。绝大多数人并不知道一天释放1000千克的碳到底是很多还是很少。排放一小时高密度的颗粒物是否正常？这些问题都成为PEIR可视化设计的考虑因素。但是，必须记住一点，虽然结果数据不是可以马上被理解，但是它们都是通过地理位置数据衍生出来，而地理位置数据本身是非常直观的。可能很少有如一个人在物理空间的地理位置这样易于被立即理解的数据。因此，我们使用地图作为数据可视化的锚点，从地图开始着手。

映射基于地理位置的数据

基于地理位置的数据驱动着PEIR系统，因此交互式的地图是用户界面的核心。我们最初使用GMap API，但很快就由于灵活性原因采取了另一个方案——采用Modest Maps取而代之。Modest Maps是基于“贴图”(tlebased)的地图的展现和交互库，它以Flash方式展现，通过ActionScript 3.0实现。Modest Maps提供了一组核心特性，如放大缩小，而且设计师和开发人员可以很容易地定制展现方式。Modest Maps的实现方式可以容易地转换地图“贴图”，这些地图“贴图”可以是微软的，定制的或者以上全部包含。我们可以自由地调整颜色、布局和全局样式，这些带来了良好的设计实践和有用的可视化展现。而且由于其灵活性，我们可以在地图上结合或者补充自己的可视化展现方式。毕竟，我们不想把自己仅仅受限于地图，Modest Maps为我们提供了这种灵活性。

通过视觉提示(vsual cue)进行实验

在决定最终的映射机制之前，我们实验了多种不同的方式来呈现PEIR数据。在设计过程中，我们考虑了几种参数：

·如何使用户能够马上和各种旅行轨迹进行交互而不会把地图弄得很混乱？

·如何能够同时呈现静态（用户处于悠闲状态）和运动（用户在运动）的数据块？

·如何呈现从所有四个微观环境模型中获取的数据值？

·应该采用什么颜色来表示GPS轨迹、影响和排放？

·如何把重点转移到实际的数据上，而不是底层的地图“贴图”？

映射多元地理位置轨迹

在设计过程的早期阶段，我们把GPS轨迹以用户通常看见的地理位置轨迹的方式进行映射：只是简单地一条线从一个点到达另一个点。在考虑微观环境模型的值之前采用这种方式，因此地图是通过使用Modest Maps和OpenStreetMap的“贴图”数据的方式做的简单实现。其GPS轨迹是单一色的，只唯一表示地理位置信息；在轨迹的最后有个圆圈(crcle)，因此用户可以知道旅程是从哪里开始，然后在哪里结束。

这在某个范围内是可行的，但是很快我们就需要对更多的数据进行可视化，因此我们改变了呈现方式——基于影响和排放值来对轨迹进行着色。其着色方案采用了五种不同深度的红色。碳影响的级别越高，其红色就越深。同样地，碳影响级别越低的旅程，其红色就越浅。

这种呈现方式的隐含意思是用户对环境的影响越大，该旅程在地图

上的显示就越突出。这种实现方式的问题是在地图的轨迹并不明显（见图1-1）。我们尝试过使用更鲜亮的颜色，但是鲜亮的颜色和地图上现存的颜色交杂会显得比较混乱。虽然我们希望轨迹显示上能够很明显，但是我们不希望使用户产生视觉疲劳。为了解决这个问题，我们尝试了一种不同的映射机制，该机制采用的方式是在地图上的所有旅程都用单色显示，但是使用了一些圆圈对影响和排放进行加密。所有的轨迹都是白色的，模型值通过圆圈实现可视化表示，这些圆圈在旅程的终点展现，其大小不一。模型值越大，圆圈的面积就越大；反之，模型值越小，圆圈的面积就越小。但是这种设计策略很快就被淘汰了。

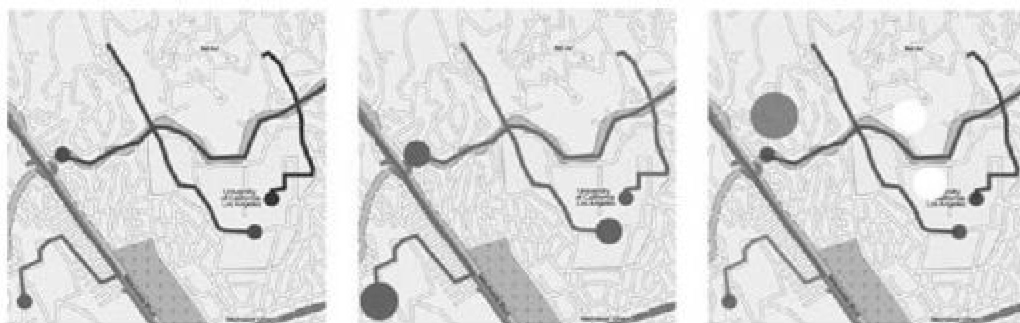


图 1-1：我们在地图上对不同的视觉提示进行实验，通过影响和排放值来最佳展示地理数据。以上展示了我们初始设计的三次迭代。最左边的地图显示根据碳影响所呈现的彩色GPS轨迹；对于中间的地图，我们通过单色的圆圈面(aea circle)来表示影响；对于右边的地图，我们结合了GPS数据来呈现用户什么时候有空，然后重新采用单色的颜色编码方案（见彩图1）

只在轨迹的终点（通过圆圈）来表示值，这种显示方式的问题之一是用户会误认为圆圈意味着每次旅程的终点发生了一些事情。但是，实

际上并不是这样。地图本应该展示的是在旅程的全部过程中发生了一些事情。

在你旅程的任何地方都释放碳，而不是收集起这些碳然后再在终点被释放。我们切换回原来的对旅程进行颜色编码的方式，删除表示模型值的大小不一的圆圈面。在设计过程的这一方面，我们现在有两种类型的GPS数据：动态的和静止的。动态的GPS数据意味着用户在运动，不论是徒步行走还是驾车行驶；静态的数据表示用户没有在运动。她可能坐在桌子上或者被交通拥堵困住。为了展示静态的数据块，我们并没有完全抛弃在地图上使用圆圈面的方式。圆圈越大，意味着历时周期越长；相反地，圆圈越小，意味着历时周期越短。和旅程相似，也是通过线条来表示的，圆面相应地进行着色。例如，如果用户选择通过颜色来标示颗粒物排放值，在高速公路上静止的数据块会显示为颜色鲜艳的圆圈。

然而，我们还是面临着与之前相同的问题：试着使轨迹能够在地图上突出显示而且不会和地图已有的颜色相冲突。我们已经试着为这些轨迹采用不同的颜色机制，但是尚未尝试改变实际地图的色调。Trulia Snapshot是为房地产做地图的公司，受到它的启发，我们用灰色来表示地图“贴图”，对换颜色过滤器，这样把原来颜色很浅的地图元素的颜色变深，反之亦然。更具体地说，地形原本是浅色的，那么现在它应该是深灰色的；而马路原本是深色的，现在改为浅灰色。这种深色调的地图能够突出显示浅色的轨迹，而且由于地图是灰度模式图，颜色冲突变少

了（见图1-2）。用户不需要费很大的劲儿就能把他们的数据从道路和地形地貌中区分出来。Modest Maps提供了这种灵活性。



图　1-2：在当前的映射机制中，我们采用颜色过滤器来高亮显示数据。地图仅仅是提供上下文信息。链接的直方图显示映射后的数据的影响和排放分布。当用户通过滚动条看直方图的某一栏，相应的GPS数据会在地图上高亮显示（见彩图2）

选择一种颜色机制

一旦我们把地图“贴图”通过深色背景显示，而把旅程通过浅色前景显示，我们确定了需要使用什么颜色。这一点很重要，因为用户会识别一些颜色表示某些特定事件。例如，红色通常表示停止或者前方有危险，而绿色则意味着进展或者增长，尤其是站在环境的立场上看。

另外很重要的一点是不要使用太多颜色对比非常分明的色彩。采用颜色很不相似的不连续的数据则意味着是分类数据。但是，模型值有连续的范围。因此，我们采用有稍微渐变的颜色。在早期的设计版本中，我们尝试使用不同深度的绿色色阶来表示。用户评论说因为绿色通常意味着好的方面或者是对环境友好，通过绿色来表示影响和排放程度很高

会显得很怪异。于是，我们采用了不同色阶的绿色和黄色组合来取代。从低值到高值，我们增量式地把颜色分别从绿色切换到黄色。把影响或者排放值为零的旅程显示为白色。

使旅程有交互性

用户一次可以潜在地映射成百上千的旅程，提供关于旅游习惯、影响和排放的一个总体概要，但是用户也需要查看每个旅程的细节。仅仅把一个旅程映射到地图上是不够的。用户需要能够和各次旅程进行交互，从而能够清楚自己相关旅程的环境信息。

当用户在PEIR地图上通过滚动条来查看旅程，该旅程会高亮显示，而所有其他的旅程都会变得暗淡，和背景融合起来但又不是完全消失。更确切地说，对感兴趣的旅程的透明度减少了，而其他旅程的模糊度变成原来的5倍。Cabspotting是对旧金山的出租车活动做的可视化地图，是它启发了我们实现该效果。当用户在地图上点击一个旅程，该旅程的日志会自动滚动到用户感兴趣的旅程上。再一次强调，该设计的目标是为了给用户提供尽可能多的环境信息，而且不会显得整个屏幕很杂乱。

当然，这些特征只在某个范围内处理多个旅程。例如，如果在一个密度高的地区有成百上千的长途旅行，这些旅程会由于杂乱而变得难以分析导航。因而我们结合用户提供的元数据如标签和分类，期望提高这一领域。

呈现分布

PEIR在地图右侧提供直方图来显示对选择的路程的影响和排放分

布。有四个直方图，每个直方图都表示一个微观环境模型。每当用户从旅程日志中选择旅程时，直方图就会自动更新。如果在绝大多数情况下，旅程在影响和排放上的值很高，直方图就会向右倾斜；类似地，如果在绝大多数情况下，旅程在影响和排放上的值很低，那么直方图就会向左倾斜。

我们最初认为直方图会是有用的，因为它们在统计上应用得如此广泛，但实际情况并非如此。直方图给人带来的更多是困惑而不是对事物的洞悉力。虽然只有很小的测试组认为它们是有用的，绝大多数人会认为横轴是时间，而纵轴是影响或排放值。人们似乎对随着时间变化的模式更感兴趣，而不是对全局分布。因此，我们转为采用基于时间的条形图（见图1-3）。用户能够看到它们随着时间的推移造成的影响和排放的连续值序列，而且可以按周浏览。

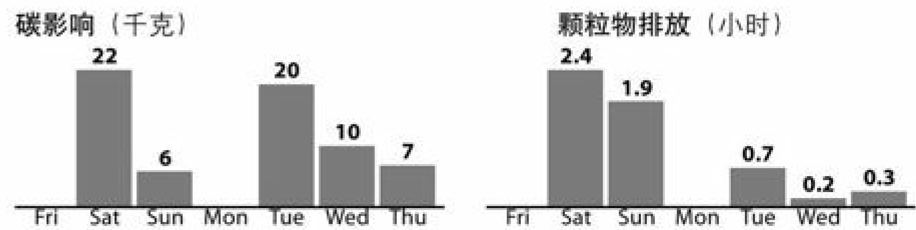


图　1-3：实践证明：时间序列条形图比基于值的直方图更有效

分享个人数据

PEIR允许用户和他们的Facebook朋友互相分享自己的影响和排放值，这种方式可以作为另一种比较有价值的方式。正是通过分享，我们可以绕过轴线的绝对范围解释，而是把重点放在相对值上，这样可以进一步帮助我们进行推导。虽然1000千克的碳看起来可能很多，但是和其

他用户进行比较后，可以改变这种误解。和其他的Facebook朋友相比，我们的PEIR Facebook应用显示了用户在Facebook的个人信息的汇总值（见图1-4）。

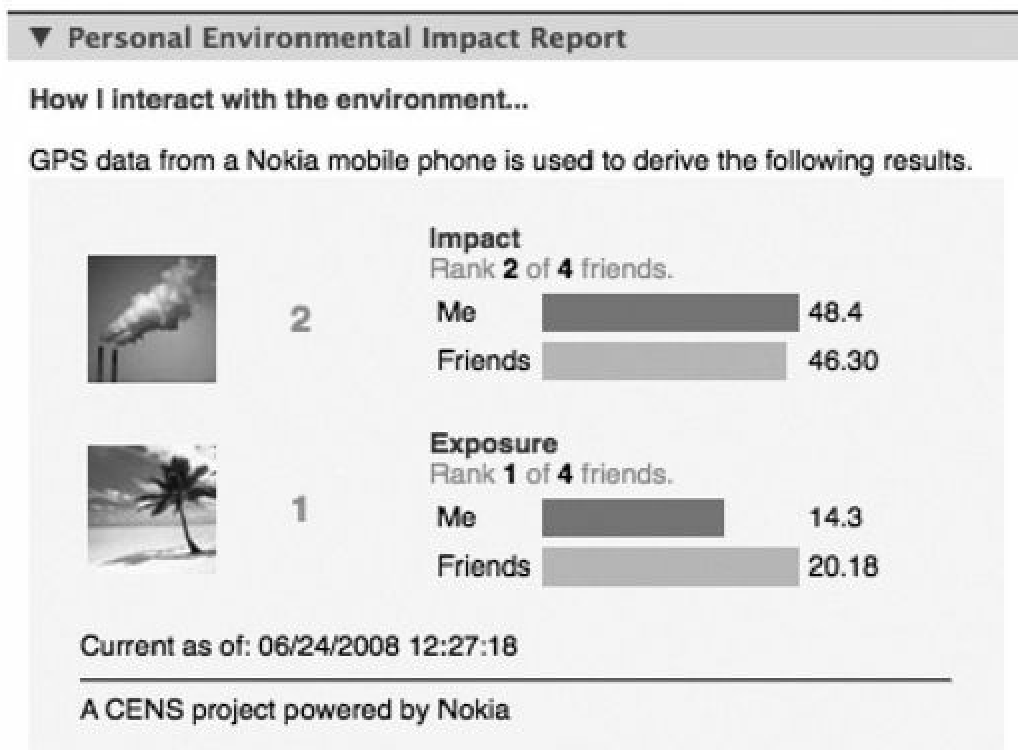


图　1-4：PEIR的Facebook应用允许用户分享他们对影响和排放值的发现，以及与朋友的进行比较（见彩图3）

PEIR Facebook应用显示了用户的影响和排放值的条状图，以及他或她朋友的影响和排放值的平均值。该应用也显示了这些值的全局排序。影响或者排放值越低的用户，排名越高。图标也提供了更多的环境信息。如果影响很高，则会显示一个带有散发着很多浓烟的烟囱图标；如果影响值低，则会显示一条溪流，溪流里倒映着蔚蓝的天空。

把注意力重新转回PEIR的用户界面，除了个人信息以外，用户也有一个网页。该网页显示上周用户影响和排放值的排序，也显示了用户好

友的排序。其目的是为了鼓励用户减少影响和排放，提高排名，同时也鼓励他们的朋友这么做。虽然通过千克或小时为单位表示的影响和排放的实际值可能最初是不清晰的，但排序的用处可以立竿见影。当用户追求更高的排序，从PEIR微观环境模型值来说也意味着更多，当打视频游戏也采用相同的方式时，其分值也开始有了意义。

读者应该注意到其中没有分享任何GPS数据。我们对于隐私数据非常严谨，采取了很多措施使得某些数据是私有的，这就是只有影响和排放的汇总值在网页中显示的原因。

YFD

PEIR处理的是一些不是立即相关的数据，而YFD则刚好相反。YFD帮助用户跟踪每天日常对话的数据。和PEIR一样，YFD的目标是使得一些生活中的小事更加明显。正是小的选择反而产生了巨大的影响，可视化可以证明这一点。

我们回到之前提及的一个挑战。我们希望用户能够频繁地用Twitter对话，把个人的数据收集转为他们每天的Twitter日常习惯。数据收集背后的动机是什么？为什么用户要跟踪他们自己的饮食或者睡眠习惯？可能有些人想要减肥是为了在异性面前可以更自信，或者是想获得更多的睡眠是为了不在办公桌上睡着。但是，可能有另一个用户想增加体重，因为她生病时体重减轻了，或者是她睡得太多，每天起床经常感觉有气无力。其他用户可能仅仅是好奇。很明显的一点是，无论其动机是什么，每个人对于个人数据收集都有他自己的理由。作为对用户的提醒，

YFD重点突出了该动机，因为不论一个人正在尝试的饮食和睡眠系统如何，除非他们自身真正希望改变，否则人们是不会有变化的。注意图1-5的屏幕中间通过大号字显示的个人动机。

需要注意的另外一点是：每个跟踪页面的最上方显示了最近最经常发生的事情。这么做有以下目的：首先，每当用户发送一个Twitter消息数据，就会被更新，因此每当用户登录到YFD时，就可以看见自己的状态。其次，我们不希望用户使用起来，体验和Twitter本身差别太大，还是进一步使用户把YFD的Twi tter消息作为Twitter的日常习惯。最后，YFD的设计方案绝大部分是源于PEIR的经验。用户似乎期望的是基于时间的可视化，因此绝大多数的YFD可视化就是基于时间来呈现。

有一个特例是对于感觉和情绪(felings and emotions)的跟踪（见图1-6）。正如每个人都会告诉你，情绪是无比复杂的。如何量化幸福、伤心或者焦虑？把情绪分解为图形和数字看起来是不对的，因此，我们采用有序的标签云图(srted tag cloud)取而代之。后者在某种程度上，让人感觉更有生命。出现频度更高的情绪比频度低的在呈现上字体格式更大。YFD跟踪器在开发的早期都是模块化的，但是我确实计划最终结合所有的跟踪器，把YFD当作用户生活的仪表盘。感觉跟踪器会在所有跟踪器的中间。最后，我们做的每件事都是由我们的感觉或者我们想要什么样的感觉来驱动的。

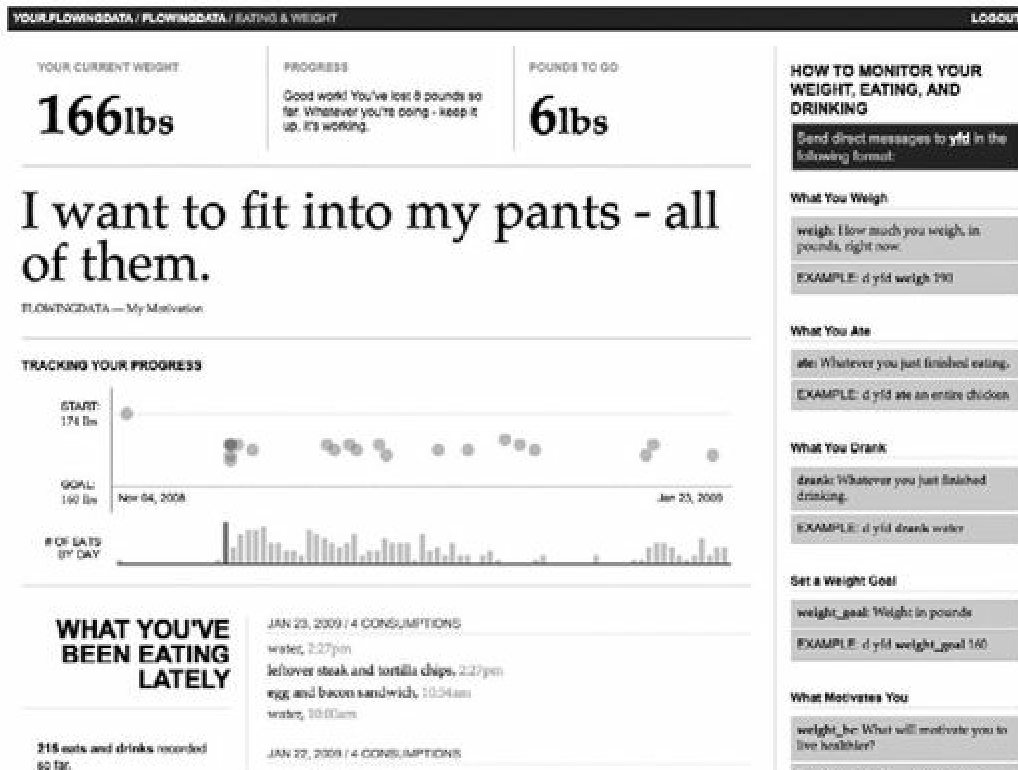


图　1-5：人们为不同原因跟踪他们的体重和饮食。YFD把这些动机作为用户界面的焦点（见彩图4）



图　1-6：用户还可以跟踪他们的感觉。不像其他的YFD跟踪器，情绪页面不包含任何的图形图表。选择单词“云”(coud)是为了提供更多

的有生命的情感可视化

要点

数据可视化通常全部都是关于数据分析和技术结果，但实际上不需要如此，尤其对于个人数据收集。人们收集自己的数据并不一定是为了追求实际数据。他们主要是对结果蕴涵的信息以及如何使用自己的数据来提高自己感兴趣。为了实现这一点，人们在可视化过程中需要观察的不仅仅是数据，他们更需要观察自己。生活很复杂，数据表示生活，用户希望能够通过数据在某种程度上能够了解这种复杂性。但这并不意味着我们需要简化数据或者信息。相反地，我们使用数据可视化来教授知识，吸引用户的兴趣。一旦用户产生了兴趣，我们可以为用户提供某种方式，从而可以更深入地钻研、探索他们的数据；或者更确切地说，探索和理解其在数据中的生活。这依赖于统计学家、计算机科学家和设计师来合理地讲述其中的故事。

如何参与

PEIR和YFD目前只能通过邀请的方式加入但是如果你希望参与，请访问我们的网站，分别是http：//peir.cens.ucla.edu和http：//your.flowingdata.com。此外，如果你想和PEIR研究组一起协作结合新的模型、策略或数据可视化，或者如果你对于如何提高YFD有自己的见解，我们都非常期望听到你的意见。

第2章　美丽的人们：设计数据收集方法时牢记用户

Jonathan Follett和Matthew Holm

简介：用户共鸣正当其时

始终牢记受众的期望和需求。这个指导着用户体验(uer experience,UX)设计领域的原则，有目共睹——足够引起任何创造新的、创新性的数字技术或者正在改进现有系统的任何专业人员的注意。“当然有人在用该产品！”虽然以用户为中心的设计过程存在很大益处，比如增强了产品可用性和客户满意度，减少了800服务呼叫，但是这个看似简单的建议人们并非总是遵守，尤其是当涉及数据收集方面时。

什么是用户体验

用户体验是一个新兴的、跨学科的领域，主要集中在设计易于理解和使用的产品和服务上。它主要关注于使得系统能够适应并服务于用户，而不是使用户适应并服务于系统（见图2-1）。用户体验专业人员可以包含以下方面的从业人员和研究人员：可视化设计、交互设计、信息架构、用户界面设计和可用性。用户体验领域与人为因素以及人机交互密切相关，它和人种学以及心理学也有一定联系：用户体验专家是用户的倡导者。通常，用户体验设计技术应用于计算机桌面软件和分布式Web软件。不过，支持者可能会更广泛地使用这一术语，用其描述任何复杂的设计经验，比如博物馆展览或者零售店的用户访问设计。

把用户体验的最佳实践应用于数据收集的好处

当涉及数据收集时，用户体验设计尤为重要。数据（最宝贵的数字

资源）来自于人们及其行动。因此，设计人员和开发人员应该一直考虑这些人，不仅仅是考虑他们想要收集的数据。收集在线人们的数据的关键方法当然是通过令人厌恶的表单。对于业务来说，没有比表单更有价值的方式；而对于参与者而言，也没有比表单更令人厌烦和无聊的方式。

作为用户体验的从业者，我们经常处理借助Web表单从大量的受众那里收集来的数据。正如我们所见，优雅的视觉设计的表单一次次极大地帮助了数据收集。对于任何表单设计项目，其所提出的挑战是：虽然数据收集比较简单，但要收集到有用的数据却可能比登天还难。表单设计很重要（见图2-1），它会直接影响到你获取数据的质量：设计良好的表单可以收集到更准确和更相关的数据。

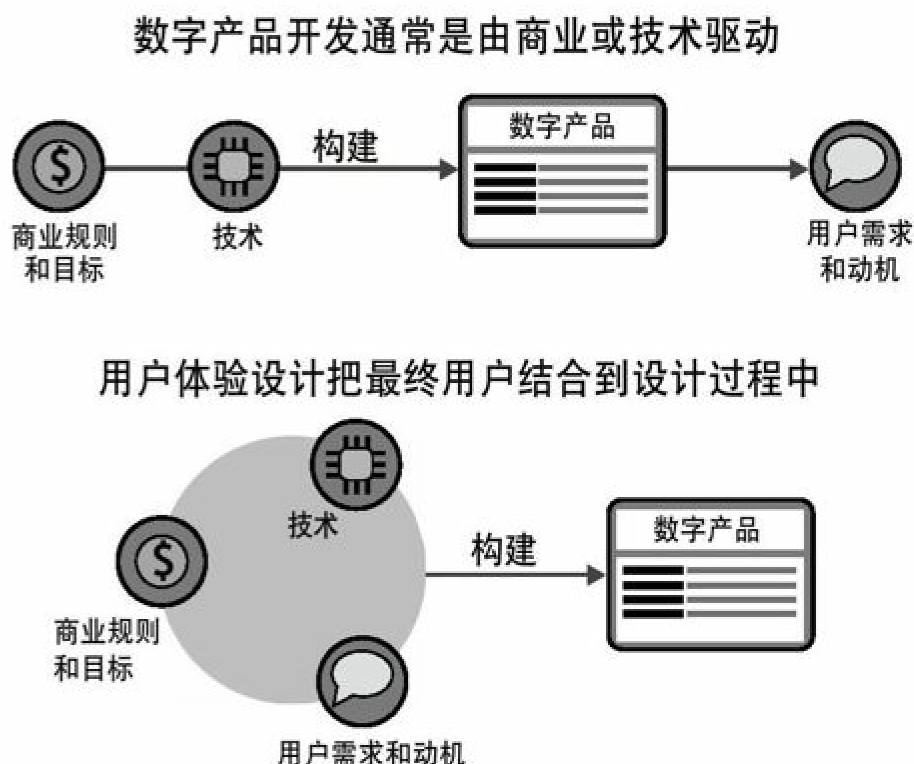


图　2-1：用户体验设计过程不是把受众的需求作为事后反思，而是在设计阶段解决处理受众需求、业务需求和技术可行性

那么，到底是什么驱动人们去填写表单、创造我们需要的数据？作为设计开发人员，我们如何才能促使他们更切实、有效、准确地完成这件事？

我们将分析一个案例研究，用来说明一个通过用户体验的最佳实践和原则来提高自主自愿的问卷回答完成率的简单的表单设计例子。

项目：关于一个新奢侈品的用户调查

该项目是为一个营销咨询公司Urban Wallace Associates做网上调查，其目的是探测消费者对一种新的奢侈品的兴趣。（为了保密，在本章中，我们不得不改变调查问题内容的一些细节。）被调查的群体与该产品的最终零售受众群体是同一群体：年龄在55～75岁的有钱人。

电子邮件调查不是我们的客户的第一选择。Urban Wallace Associates已经尝试过对目标群体进行电话调查。“通常情况下，我们打电话时，大约35%是录音电话，”UWA主席Roger Urban说，“但是在有钱人群体中，有超过80%是录音电话。即使有人接电话，接电话的人也通常是管家！”

无法通过电话获得令人满意的目标受众的回答，我们的客户转而使用电子邮件。他们选择这种通信方式的原因是，对于富裕的群体，电子邮件是普遍使用的工具。虽然电子邮件面临其自身的“守门员”(gtekeeper)（即自动垃圾邮件过滤）但很少有人雇佣别人为自己读取邮件。即便是有钱人，他们也仍然是亲自打开自己的电子邮件。

Urban Wallace Associates争取到一个电子邮件营销公司为他们生成和预审收件人列表，并提供投放和跟踪外放信息。我们公司的任务是设计和构建该调查需要的“着陆页”(lnding page，即访问的起始页面），当收件人点击邮件正文的一个链接，该页面就会在他的浏览器打开，然后收集信息到数据库。我们在这项调查中的首要任务是维护并保持问卷调查

的Web页面的令人愉悦的氛围，因而受访者能够更愿意填写表单。我们的另外一个任务是为客户创建一个简单的页面，这样当有数据进来时，他可以审查实时的报告结果。

数据收集面临的特殊挑战

数据收集提出了一些特殊的挑战，包括可访问性(acessibility)、信任和用户动机。以下各节将讨论这些问题如何影响了我们的设计。

可访问性挑战

倡导Web的可访问性（设计使得网页和网站对于有特殊需要的人和残疾人仍然是有用的）通常情况下，设计一个具备可访问性的站点同时也应是创建一个对于每个人都更易用的网站。这并不只是该案例下的理论想法，因为目标受众是接近退休或者已经退休的成员，与年龄有关的视力障碍将成为真正需要关注的问题。约72%的美国人在45岁左右时报告存在视力障碍。

关于年龄问题的另一方面（由于担心表现出歧视性，很少被谈及）是老年人使用计算机和互联网的人数较少，而且不像那些和计算机一起成长的年轻人那样能够对它们驾轻就熟。（收入越高的个人，通常使用计算机和网络越多，因而，这种和年龄相关的影响在我们的受众群组中被削弱了。）被一个设计不清晰的调查所僵住的受访者不太可能会提供准确的信息——或者更确切地说，根本无法完成这项调查。就我们而言，对于所有的这类项目，回顾那最基本格言是值得的：了解你的受众。

直觉的挑战

可访问性是个功能上的问题（受访者如果不能读懂一个调查，就无

法完成该调查问卷），我们的项目面临着在本质上更情感化的其他挑战，而且依赖于受访者如何理解提问者及其问题。

建立信任

Internet用户很清楚在线透露信息可能会有严重的后果，包含增长的垃圾信息、电话诱惑和垃圾邮件，以及欺骗和身份盗窃。因此，对这些想做网络市场调研者来说，建立信任是很重要的因素。即使对产品和调查的答复整体非常肯定（我们将在后面详细介绍），如果问一些参与者为什么对产品不感兴趣，有如下的回答：

“不信任你们公司”

“从未听说过该产品的提供者”

“不相信该产品声称的任何方面”

“买不起……不信任……好得令人难以置信，因此很可能不是真的。关于该产品，请不要再联系我！”

这些受访者的答复，说明了为建立网络信任，我们还必须走很长的路。更重要的是，对于这个项目，明确要求我们不能销售任何东西——我们是在做调研。我们的客户Roger Urban一开始就告诉我们：“我不想听任何像是搞销售的话。”明确地提供关于Urban Wallace Associates的信息是很必要的，这样人们就知道是什么样的公司在对他们做问卷调查；而且明确地指出我们不会收集他们的个人资料，而且也不会再骚扰他们。其中唯一的障碍是，我们的客户调研需要知道每个答复者在美国哪个州居住。因此我们需要探索出一种合适的方式，在没有违背我们正在

尝试建立的信任的宗旨的情况下捕获该信息。

调查的“度”

避免受访者不积极参与是我们所面临的最大困扰之一。客户和我们需要在早期达成一致，保持调查页面只通过一个屏幕就能够完全显示。如果需要多个屏幕显示，不但需要受访者有更多的耐心，而且可能需要他们执行一些额外操作（比如点击“下一个问题”按钮）。一个调查在任何时候需要受访者多执行一步操作，就相当于“邀请”他决定这额外的付出是不值得的，然后选择放弃。此外，我们希望在任何情况下，调查从始至终都应该避免恐吓受访者。多个屏幕或者在一个屏幕上看起来有太多问题，这些都增加了受访者中途退出调查的可能性。

准确的数据收集

在本次调查的设计阶段，我们考虑的一个特别重要的问题是收集的数据必须尽可能准确——可能该数据只是一个显而易见的表述，但要保证准确性，还是很困难。我们的表单设计需要参与者诚实的回答，而且其回答不受其他因素影响，比如说，潜意识地希望能够让提问者高兴（这是这类调研的“通病”）。收集人们的意见这方面的数据和收集信息之间的区别是，后者可能本质上更偏重于管理实施，比如配送地址；配送地址可以很容易验证，但观点意见这类数据，本身就是主观的，通常不是那么可靠。虽然在本章中不会深入涉及设计民意调查和评测结果数据这门科学，但是我们将会讨论为鼓励受访者填写准确的答案，我们团队所采用的一些语言以及做出的其他选择。

动机

最后，虽然我们已经谈到了如何让受访者有可能使用表单，以及关于以下方面的诸多问题：让受访者信任我们、参与调查；避免问题太多“吓跑”了受访者；确定我们没有潜意识地影响他们的回答；以及我们之前尚未提及的一点，而且可能是任何调查最重要的部分：受访者为什么要参与？对于这类调研，不存在任何参与的受益动机，不像Amazon的Mechanical Turk一样的在线论坛，对于后者，用户可以在空闲时间完成任务，而且每项任务可以获取几美元或者几美分。然而，当对方没有明显的利益时，如何说服一个人花时间来回答你的问题？

设计解决方案

我们已经谈到了对于数据收集项目的一些内在的陷阱，在后面几章将分析我们设计的具体细节，包括排版、Web浏览器的兼容性和动态表单元素。

设计哲学

当我们是为了获取受访者的答复而进行设计时，从用户角度来制定问题是很重要的。项目设计时很容易因为技术限制而陷入其中，从而是为计算机设计而不是为用户。但是表单数据是由用户主动生成的（而不是通过传感器或其他输入被动生成），而且要求参与者决定如何以及是否回答你的问题。因此，以何种方式来收集参与者的数据至关重要。我们为该项目设计Web表单时，重点在于对调研参与者和客户商业目标的动机进行权衡。客户的主要商业目标是：收集数据，决定目标受众是否对购买一款新的奢侈品感兴趣——与以用户为中心的设计理念一致。通过把个人同时置于顾问和潜在的未来客户这两种核心角色中，业务目标为以用户为中心的设计决策提供了强有力的理由。

以下是我们在制定设计决策时遵循的一些指南：

尊重用户

为了使设计在整个过程中保持以人为本，需要了解考虑用户的情绪反应。为了说服他们参加，我们首先必须对他们表示尊重。他们不是傻瓜，而是我们的潜在客户。我们本能上都清楚这些，但令人惊讶的是，

我们会多么容易忘记该原则。如果我们对用户表示尊重，自然地就会希望构建的数字产品对他们来说是可访问的、可用的且易于理解。这种观点影响了我们做出的每个选择，包括从语言、布局直至技术。

展现真实的人们

对于时间节奏要求很快或者预算受限的项目，我们通常没有资源基于目标市场研究来塑造一个完整的用户个人信息或角色，或者在用户的工作环境中观察他们。在这些情况下，采取简单的“游击队”式的用户体验技术使用户产生共鸣，其指导性的设计决策是想想一个我们认识的真实的人，我们真正地想要帮助他。我们已经有一些这样的角色替身来指导我们思考，包括年迈的父母、一些很了解其先前情况的商业伙伴。当然，通过数字产品去想象这些人只是设计的第一步。既然对他们很了解，我们也可以在设计的初期测试时取得他们当中一些人的帮助。

最终，人们会调整自己的行为，如果有必要，那么他们的行为几乎可以和任何设计保持一致。用户体验的目的是优化这些设计，使得人们会期望使用某种产品或服务，而且可以不需要调整他们的行为，就能够更方便、快捷地使用这些产品和服务。

表单布局设计

通常来说，不论表单做得多么漂亮，它都不太可能会达到取悦用户的水准。设计师没有“圣杯”使人们对填写表单充满激情。但是表单的审美艺术还是很重要：清晰的信息、可视化的设计可以明晰地引导用户看哪些方面，鼓励他们对表单填写坚持到最后而不是中途放弃任务，从而

减轻用户的无聊厌倦情绪。好的表单设计不会把用户注意力吸引到自己本身，表单应该几乎是不可见的，永远信守其主要目的——从人们那里收集准确的信息。虽然表单设计需要给人感觉既舒服又专业，但在绝大多数情况下，和其他形式的Web页面相比，经过合理的视觉处理的表单会给人感觉既含蓄又实用。表单视觉设计只能通过它如何有效地促使用户完成任务这个指标来判断。对于这个项目，我们在设计上的努力主要集中在排版、页面布局和交互设计这三个领域。

Web表单排版和可访问性

通常来说，年纪较大的读者看较小的东西很吃力，而且调查参与者也不会“慷慨”到为了阅读一个表单，甘愿让自己的眼睛受累。由于调查项目的目标受众是年龄较大（55～75岁）的群体，我们清楚整体的易读性将是个问题。

我们采用Sans Serif字体类型Arial（类似于微软推出的骨干字体Helvetica)，它对几乎100%的Web浏览器都是标准的，我们对标题和正文的字体大小分别设置为20像素(pxel)和14像素。虽然字体大会造成整个页面稍长，但由于其带来的易读性方面的改善，该代价是值得的。

行间距不是很紧，且采用左对齐的方式。每行长度大约是85个字符。对于大多数文本，我们设置了高对比度——白底黑字的显示方式，这种设计也是出于易读性的考虑。虽然我们确实可以在策略上使用鲜艳的颜色对页面进行高亮显示，而且突出显示主要标题，但是我们并没有依赖这种方式来为用户提供额外信息。我们这样做的原因是，大约有

7%～8%的男性受众是某种类型的色盲。

给人们一些空间

没有“呼吸”空间、设计过于紧凑的表单必定会吓坏用户。因此，在布局时留一些开放空间很重要。

在我们的调查中，表单的第一部分包括对该奢侈产品的文本描述，在该文本描述中，我们请求参与者对它进行阅读和评估。由于阅读Web页面的读者的注意力能够集中的时间很短，而且他们倾向于略过一部分文本而不是通读。因此，遵从Web写作的最佳实践，我们把250个单词的产品描述切分为包含多个标题的子部分，即抽取出关键点，并把这些点划分成几个易于理解的不同分块（见图2-2）。



图　2-2：为易读性而设计（见彩图5）

适应于不同的浏览器，并测试兼容性

为了确保表单对于受众是可用的，我们设计了表单页面，这样对于

从800个像素到更多像素的不同大小的屏幕，用户都可以很容易地查看该表单。为了实现这一点，我们把表单放在浏览器中间，使用中性的灰色背景来填充由于宽频显示器引起的左右空白剩余空间，而且保证表单看起来不会显得空洞和“飘忽不定”。为了确保主流表单看起来不错且能够正常工作，我们还在所有的主流Web浏览器上进行测试，包括传统的IE6浏览器。

交互设计考虑：动态表单长度

动态表单可以“舒缓”需要回答很多问题带来的“重击”。使用JavaScript或者其他方法允许表单基于用户输入，做一些微妙的修改或加长，从而营造一种柔和的展现方式（见图2-3和图2-4）。这些技术允许我们对以下两方面做出权衡，一方面不会由于表单太长而吓坏了用户，另一方面不会使用户因为在表单长度方面感到被“欺骗”而动怒。

对于我们的项目，实际上是根据用户回答的每个问题来构建了整个调查。我们使用了一段简单的JavaScript脚本，确保每个新问题都是基于前一个问题的回答。这种解决方案的思想来自于我们当时为之工作的另一个Web站点。那个项目是为设计者创建组合站点，我们使用JavaScript来隐藏和显示不同项目的细节，使得用户一眼就可以理解设计者的所有作品，然后对感兴趣的领域进行深入研究，而且都不需要离开该站点的首页。这种思想不但没有把用户淹没在过多的信息中，同时还使得这些信息能够很快地被用户所访问。我们在设计该调研时，一直牢记着该想法。以下是我们使用的代码：

```
<script language="JavaScript">
//This finds the word"Yes"in an input value and displays the
designated
hiddenElement
(o hides it if"Yes"is not found)
function switchem(switchElement,hiddenElement){
if(switchElement.value.search ("Yes") >-1)
document.getElementById(hiddenElement).style.display='';
else
document.getElementById(hiddenElement).style.display='none';
}
</script>
<script language="JavaScript">
//This finds the word"No"in an input value and displays the
designated
hiddenElement
(o hides it if"No"is not found)
function switchem2(sitchElement,hiddenElement){
if(switchElement.value.search ("No") >-1)
document.getElementById(hiddenElement).style.display='';
else
document.getElementById(hiddenElement).style.display='none';
}
</script>
```

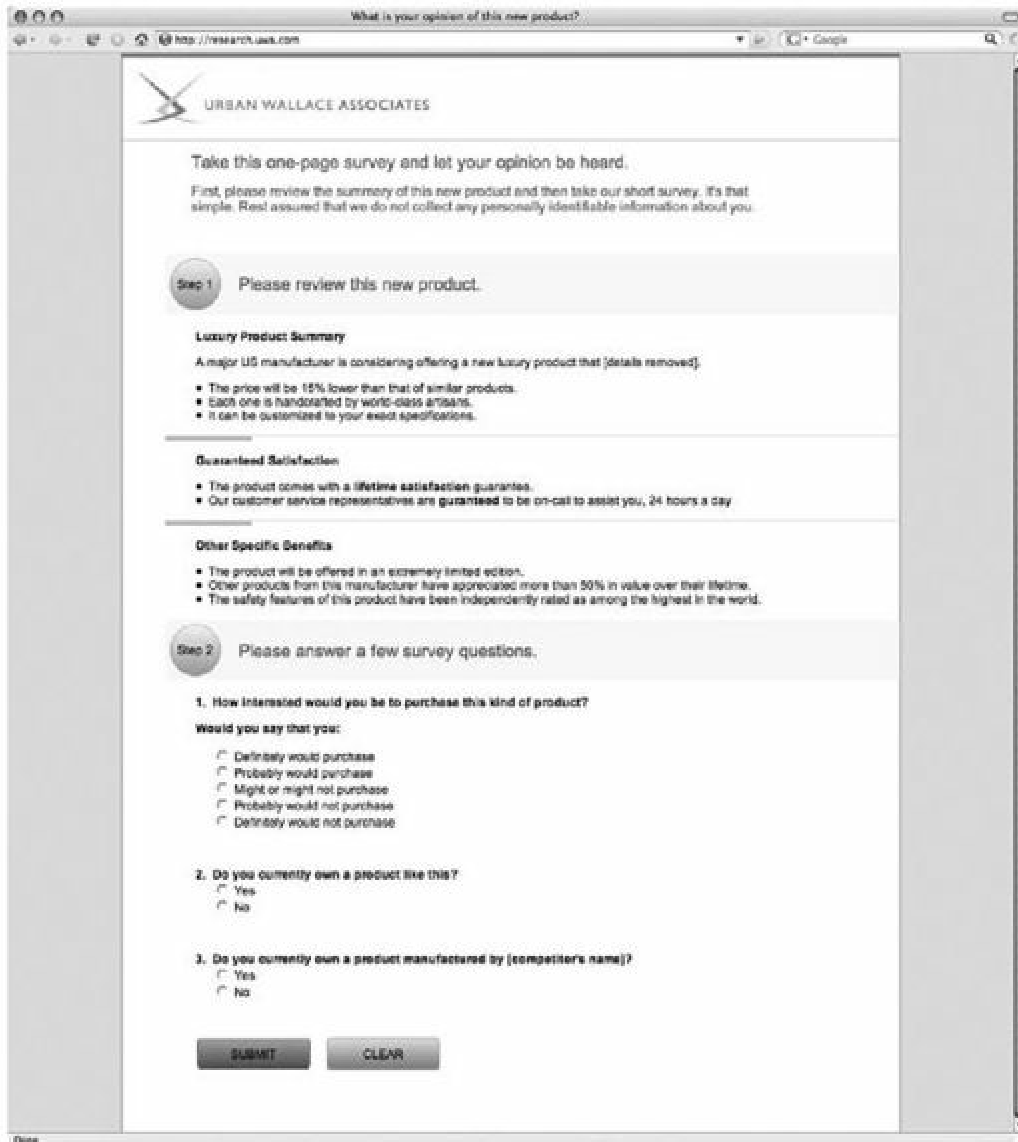


图　2-3：调查开始只有3个问题（见彩图6）

```
……
<li id="survey1"class="surveynum">How interested would you be to purchase this
kind of product?
<p><b>Would you say that you: </b></p>
<ul class="nobullet">
<li><input
onclick="switchem(this, 'survey2') ; switchem2(tis, 'survey3') ; document.
get Element
ById ('surveytextarea') .value=''"type="radio"name="q1"value="Yes,
Definitely would purchase">Definitely would purchase</li>
```

```
   <li><input
   onclick="switchem(this, 'survey2'); switchem2(tis, 'survey3');
document.
   getElementById('surveytextarea').value=''"type="radio"name="q1"val
   would purchase">Probably would purchase</li>
   <li><input
   onclick="switchem(this, 'survey2'); switchem2(tis, 'survey3');
document.
   getElementById('surveytextarea').value=''"type="radio"name="q1"val
or
   might not purchase">Might or might not purchase</li>
   <li><input
```

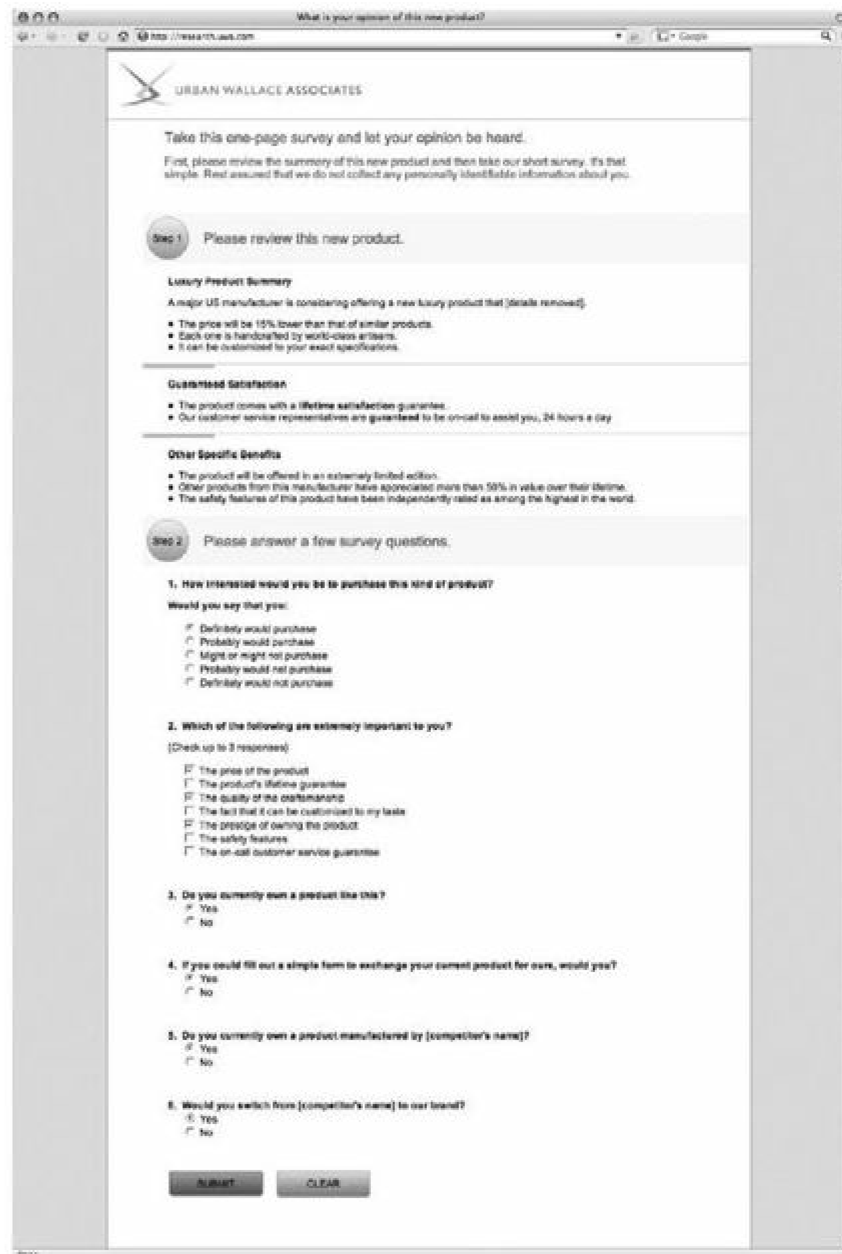


图 2-4：基于用户输入，调查可能会扩展到6个问题（见彩图7）

```
   onclick="switchem(this, 'survey2'); switchem2(tis, 'survey3');
document.
   getElementById('q2a').checked=false;
document.getElementById('q2b').
   checked=false; document.getElementById('q2c').checked=false;
document.
   getElementById('q2d').checked=false;
document.getElementById('q2e').
   checked=false; document.getElementById('q2f').checked=false;
```

```
   document.getElementById('q2g').checked=false"type="radio"name="q1'
   Probably would not purchase">Probably would not purchase</li>
   <li><input
   onclick="switchem(this, 'survey2'); switchem2(tis, 'survey3');
document.
   getElementById('q2a').checked=false;
document.getElementById('q2b').
   checked=false; document.getElementById('q2c').checked=false;
document.
   getElementById('q2d').checked=false;
document.getElementById('q2e').
   checked=false; document.getElementById('q2f').checked=false;
   document.getElementById('q2g').checked=false"type="radio"name="q1'
   Definitely would not purchase">Definitely would not purchase</li
>
   </ul>
   </li>
   <li id="survey2"style="display:none"class="surveynum">Which of
the following are
   extremely important to you?
   <p>(Ceck up to 3 responses)</p>
   <ul class="nobullet">
   <li><input type="checkbox"name="q2"id="q2a"value="The price of
the product">
   The price of the product</li>
   <li><input type="checkbox"name="q2"id="q2b"value="The product's
   lifetime guarantee">The product's lifetime guarantee</li>
   <li><input type="checkbox"name="q2"id="q2c"value="The quality of
   the craftsmanship">The quality of the craftsmanship</li>
   <li><input type="checkbox"name="q2"id="q2d"value="The fact that
it can be
   customized to my taste">The fact that it can be customized to my
taste</li>
   <li><input type="checkbox"name="q2"id="q2e"value="The prestige
of
   owning the product">The prestige of owning the product</li>
   <li><input type="checkbox"name="q2"id="q2f"value="The safety
features">The
   safety features</li>
   <li><input type="checkbox"name="q2"id="q2g"value="The on-call
customer
   service guarantee">The on-call customer service guarantee</li>
   </ul>
   </li>
   <li id="survey3"style="display:none"class="surveynum">Why are you
not interested
   in this product?
   <ul class="nobullet">
```

```
<li><textarea id="surveytextarea"name="q3"></textarea></li>
</ul>
</li>
```

结果是在采用5分制的问题1的问答中，若受访者选择其中三个正面答案中的任意一个，都会显示一个清单来帮助我们更好地识别该受访者喜欢该产品的什么方面（见图2-5）。相反地，若受访者选择两个负面答案中的任意一个，都会显示一个文本区，在该文本区中，该受访者可以准确地解释为什么他不喜欢这款产品（见图2-6）。

随着编程不断进展，以上这些都是小儿科，几乎不值一提。但是站在用户角度其所带来的影响是微妙且有力的。它意味着我们可以通过对话的方式“倾听”和“答复”用户的输入。它也意味着由于表单长度造成的心理影响是很低的，因为用户最初面对的只是3个问题的调查。该调查潜在地可以扩展到6个问题，但是这些都是在用户的“着陆页”发生的，而且不需要用户离开页面，因而避免了强迫用户点击类似于“下一页”的按钮。

Step 2 Please answer a few survey questions.

1. How interested would you be to purchase this kind of product?

Would you say that you:

- Definitely would purchase
- Probably would purchase
- Might or might not purchase
- Probably would not purchase
- Definitely would not purchase

2. Which of the following are extremely important to you?

(Check up to 3 responses)

- The price of the product
- The product's lifetime guarantee
- The quality of the craftsmanship
- The fact that it can be customized to my taste
- The prestige of owning the product
- The safety features
- The on-call customer service guarantee

图　2-5：调查细节——当用户对问题1回答“Yes”时（见彩图8）

Step 2　Please answer a few survey questions.

1. How interested would you be to purchase this kind of product?

Would you say that you:

- Definitely would purchase
- Probably would purchase
- Might or might not purchase
- Probably would not purchase
- Definitely would not purchase

2. Why are you not interested in this product?

图　2-6：调查细节——当用户对问题1回答“No”时（见彩图9）

设计信任

为了尝试与受访者建立信任，我们确实做了一些具体的实践，暗示受访者这是一次合法的调查，而不是惩罚“苦旅”。首先，我们在Web调查页面的最上方很显著地展示了客户的公司标识(lgo)。该标识本身又链接到Urban Wallace Associates的主站的“关于我们”(Aout Us)页面，因此参与者可以很清楚他们在和谁进行沟通。此外，我们把调查页面放在客户的主站的子域名，而不是什么第三方提供商。

正如之前所述，我们的客户所要做的调查研究需要每个受访者在美国有居住权。但是，由于我们告诉了每个受访者“我们不收集任何个人身份信息”，所以询问受访者居住在哪里会显得很不适宜。对于该问题的解决策略是自动记录受访者的IP地址，这样可以满足获取美国地理位置的需求，而且没有违背受访者的隐私。毕竟，用户的IP数据在他或她访问任何Web站点的任何时候都被记录下来；而且在绝大多数情况下，

IP数据只能用于确定用户的网络服务提供商(IP)所在的城市，否则该网络服务提供商就是匿名的。

我们后来购买了一份廉价的IP与州的映射关系数据。根据这份数据，我们可以知道收集到的每个IP地址在美国的哪个州。虽然我们可以通过在页面嵌入脚本来访问该数据库的方式实现在收集数据的过程中实时地对数字做地理信息匹配，但考虑到所面临的实际情况，我们还是选择了半自动化匹配方式。在开始阶段，项目预算和时间表不能保证能够购买额外的服务器资源来处理该任务。但更重要的是，从用户角度考虑，进行实时匹配所带来的延迟将不可避免地成为调查过程的一部分。

虽然对我们来说，能够立即获取最终的数据可能会更方便，但它也会给我们的用户带来额外的不便。当设计数据收集时，为了满足用户需求，重要的是要考虑在调查过程中，服务器首先必须完成哪些任务，而哪些任务是可以推延到数据收集之后再完成。不要要求用户——为了你——去做或者去发现你自己可以做的事。

以上所有这些都把我们带回到本章的中心，也是建立信任的最终和最核心方面：尊重受访者。通过表示你对受访者以及她付出的时间和智慧的尊重，通过交谈对话的方式和她互动（虽然事实上所有的调查问卷都是通过对机器进行预设置来完成的），而且向她显示你一直在“倾听”她的回答（例如，不要一遍又一遍地问受访者几乎相同的问题，这会显得你没有注意听她最初的回答），这样，你将增加信任，鼓励真实的回答，而且避免受访者“溜号”。

为精准的数据收集而设计

这种讨论方式似乎显得有点“亲昵”，尤其是对于只与收集来的那些“生硬”的数据打交道的人们，而不是与生成这些数据的人们打交道。但是，所有这些以用户为中心的关注都不仅仅是礼貌问题，它对于我们实际获取到的数据的可靠性也是至关重要的。Roger Urban的公司专长于通过当面沟通、邮件、电话和类似于本案例的电子邮件调查。“对于这样的调查，”他说，“你是在处理非常少量的数据，因此数据的质量是非常重要的。”换句话说，如果仅仅是基于几百个人的回答来做出重大决策，那么这些回答最好是“优秀的”。

但是，“优秀的”的回答并不意味着是肯定的回答。毕竟，这是调研，和科学家们一样，我们想要评估的是实际情况（对于这款产品，用户是否会很关注其价格？安全性是否是他们最关心的？他们实际上对我们的服务是否满意？），并查看我们的假设错在哪里。“说服技巧对于调研是个灾难。”Roger Urban如是说。人们为了使提问者高兴，通常会下意识地通过回答他们认为是提问者所期望的答案。

无论是“隐式”地还是“显式”地采用说服技巧，都将会使得你的调研数据结果有偏差。“如果你想要一份人为地非常肯定的数据，”Urban说，“我每次都可以获取给你。”但是，如果你是在做真正的商业决策或者政治决策，这种数据又有何用？

动机

在数据收集期间，你不能使用说服技巧；而在调研最初时，你又确

实需要说服受访者参与。既然没有金钱，那么对于那些受访者，他们的动机会是什么呢？

“应该总是有一些‘好处’”，Roger Urban说，“即使该‘好处’仅仅是‘表达你的观点’”。人类是有趣的动物，生冷僵硬的金钱可能并不能驱使我们去做某些事情，反而是一些更“朦胧”的“好处”可能会驱使我们去做，比如一些很得体的“恭维”。我们都希望自己被认为是专家；为了证实我们的观点是正确的，可能就足以说服我们花时间和陌生人交流。另一方面，通过参与我们可能可以获取一些“内幕信息”、看得更远，这些想法也会诱惑我们去和陌生人交流。举个例子，参与一项可以一睹Apple的下一个i-gadget设计的调查，有哪个技术痴迷者会对此不热衷呢？

对于这个项目，我们知道目标客户是年纪较大的受众群体。为了使邮件接收者参与调查，初始邮件的语言设置是很重要的，我们团队在邮件中征求了受访者的专业知识。在我们的第一封邮件中，我们对同样多的接收者测试了两种不同的标题：

“您可以为后代塑造**产品（产品信息被删除）的形象。”

“我们正在寻求‘经验之声’。”

结果是对于第一个标题，虽然赋予受访者权利来操纵未来的方向，相比之下，显然被证明了显得更短暂易逝和过于无私（毕竟，它意味着其利益只是给后代，而不是受访者本人）；而第二个则勾起人们的自我主义，把她们的年龄变成积极的一面（“经验”）。对于第一个标题，有12.90%的接收者收到邮件后点击它，这些人当中，有16.22%完成了调

查。而对于第二个标题，有14.04%的接收者收到邮件后点击它，这些人当中，有29.5%的人完成了调查。当两周后开始第二次调查时，所有消息的标题都是“经验之声”，有27.68%的接收者打开邮件，这些人当中，有33.16%的人完成了调查。第二次邮件也发给了那些第一次没有打开邮件的人们。(Eail调查的一个秘密是：给相同的邮件列表发送邮件，第二次发送通常会和第一次发送收到同样多的答复。）

用户体验哲学上值得记住的另一方面是：对一切都进行测试。在这种情况下，甚至是对你的测试方法进行测试！当你有时间和资源时，要对不同的副本、不同的布局和不同的交互设计类型进行测试，而且这些都是基于和实际用户进行的测试。

报告即时数据结果

在我们的项目中，一个特殊的考虑是结果数据的接收者，即客户，也会成为系统的用户，他们和那些调查参与者有着显著不同的需求。

由于该项目时间紧迫，客户为了确定该产品是否普遍受欢迎，需要快速看到调查结果。对于这种应用背景，我们的解决方案是采用一个HTML页面显示数据，客户可以访问该页面，其数据通过最少的格式化进行简单排序。即时、原始的调查结果首先通过邮件列表来排序（其中每个标题都发送给两个邮件列表，年龄段分别是在55～64岁和65～75岁），然后是通过人们对第一个问题，即是否对该产品感兴趣所做出的Yes/No回答。需要说服调查参与者来参与调查并完成表单填写，而和这些参与者所不同的是，客户希望的是能够尽快看到生成的数据结果。对

于客户，速度和对即时结果的即时访问比其他任何因素都要重要。因此，客户在用户体验上反映了以下这些优先级（见图2-7）。

#	ID	Mailing	State	Q1	Q2	Q3	Q4	Q5	Q6	Q7	IP Address
1	44	1	GA	Yes, Probably would purchase	[illegible]		Yes	Yes	No		24.110.208.16
2	34	1	VA	Yes, Probably would purchase	[illegible]		Yes	Yes	Yes	Yes	24.210.16.181
3	84	1	VA	Yes, Probably would purchase	[illegible]		No		No		64.12.116.13
4	104	1	FL	Yes, Probably would purchase	[illegible]		No		Yes	No	71.53.249.200
5	23	1	CA	Yes, Probably would purchase	[illegible]		Yes	Yes	No		69.3.100.186
6	105	1	CA	Yes, Probably would purchase	[illegible]		No		Yes	Yes	68.122.97.18
7	59	1	VA	Yes, Might or might not purchase	[illegible]		No		No		72.189.200.205
8	94	1	CO	Yes, Might or might not purchase	[illegible]		Yes	No	No		75.107.23.70
9	26	1	TX	Yes, Might or might not purchase	[illegible]		No		No		76.213.6.205
10	34	1	CA	Yes, Might or might not purchase	[illegible]		No		Yes	Yes	76.83.240.29

图　2-7：在这个即时数据报告屏幕上，随着数据的流入，客户可以看到调查结果

当然，最后交付的不是这种原始数据的展示。在项目完成时，我们向客户展示的是完全可排序的Excel表格，这些表格包含我们收集到的所有数据（总共包含8个邮件列表，通过两次批量发送的方式），包括在调查时还没有生成的关于所在美国的哪个州的地理位置数据。

结论和反思

最后，我们付出的这些努力都是值得的吗？这仅仅是一个Web表单，不是吗？人们每天都在填写数百万的表单。有些人可能会认为我们根本就不需要考虑如何设计——创建一个可用的Web表单的问题早已经解决了，而且是一劳永逸的。但是你永远都不应该低估在解决最普通的设计问题上所缺乏的努力，尤其是在线的设计问题。现在绝大多数的表单和20世纪90年代的表单相比，并没有太大区别。

此外，如果有一件事是一名优秀的设计师应该清楚的，尤其是对于一名遵循用户体验规范的设计师，那就是不存在“一刀切”式的解决方案。为你的用户群体采取定制方式几乎总是会提高用户体验，而且是针对数据收集这种类型来说。

对于我们的客户的情况，从结果上看，付出的努力是值得的。我们了解到，该电子邮件营销公司以前的推广活动中，通常打开其邮件的概率在1%～2%，而我们的邮件打开概率高达4%。对于打开的邮件，通常完成点击的概率在5%～7%，而我们的则高达21%。最重要的是，对于这些执行点击操作，跳到Web页面，继续操作（例如完成表单）的概率是2%～5%；而对于我们的设计，其完成概率是29%（见图2-8）。

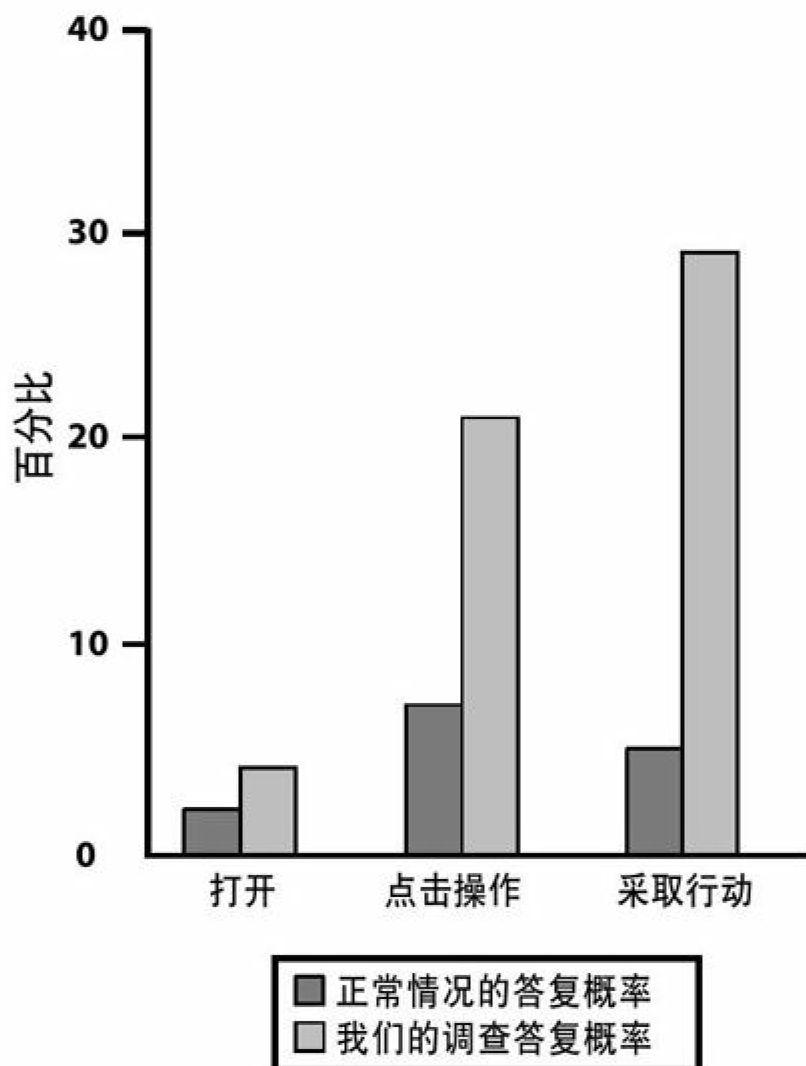


图　2-8：我们的调查答复概率要显著高于正常情况的答复概率，这归功于更好的总体用户体验

当然，这次调查比该公司之前的邮件调查收效好得多，但是也存在其他可能的原因。可能是因为该产品比该公司之前为其他产品或话题开展的调查更加吸引人，而且可能是该产品给人们带来的兴奋之情使得更多人能够参与到最后；也可能是预选的接收者比通常的要准确得多，而且这些接收者特别适合该产品；甚至可能还存在年龄上的偏见，认为是

年龄在起作用——年纪大的计算机用户比年轻的用户更有可能会打开邮件、阅读这些邮件、点击进入、完成调查，而年轻的用户是否可能更精明，而且对于不请自来的邮件更加谨慎？我们不了解对于该主题的任何研究，但它是一种可能性。实际上，虽然我们不能完全排除这些解释中的任何一个，但是电子邮件公司为我们这次调查所做的似乎并没有和他们之前为其他几百个调查所做的有什么区别。因此，很有可能可以下这样一个结论：该项目的成功和我们的表单设计有关系。

顺便说一下，虽然这和该调查的设计没有什么关系，不过也许你还是会对此感兴趣，那就是受访者对该产品本身的反馈非常积极正面。虽然客户告诉我们的是如果有10%的正面答复概率（即对第一个调查问题的回答是“Yes”），就可以推出该产品；而结果是超过16%的受访者潜在地对于购买该产品颇感兴趣。这究竟是一款什么样的产品呢？遗憾的是，保密协议禁止我们对本次调查透露更多信息。

如果你想“亲眼目睹”这款产品，你只能期望能够参与到下一次的电子邮件调查中。不要过快地把这些电子邮件扔到垃圾箱中；至少，你可以学到关于表单设计的一些好的或者差的方面。

第3章　火星上的嵌入式图像数据处理

J. M.Hughes

摘要

太空飞行器是独特的工程项目，其受到的约束和要满足的需求是地球上的任何物体所不具备的。它们必须能够承受极端恶劣的气温极限、完全的真空空间和强烈的辐射，而且其重量必须足够轻，以便可以用火箭将它们载入太空并发送到其目的地。太空飞行器是应用了极简主义的一项“运动”：只有完成任务所必需的（设备），没有丝毫多余。所有方面的设计都将测试以下几个方面：必要性、重量和成本。在发射之前，一切都要经过测试、再测试，包括太空飞行器的“大脑”——嵌入式计算机系统及其上面运行的软件。本章将概述凤凰号火星着陆探测器(Poenix lander)的图像处理软件如何获取、存储、处理图像数据以及最终如何再把图像发送回地球。

简介

人们在设计和编程实现一套嵌入式系统时，会受到很多约束条件。这些约束包括处理器速度、执行期限、允许中断的延迟时间以及内存约束等。由于太空飞行的使命，这些约束条件会变得很严酷。通常，在太空飞行器上的计算机只使用非常昂贵的能够抗辐射的内存。它的中央处理器(CU)，通常也是经过专门设计的设备，因为它需要承受高能量宇宙射线的破坏性影响。从商业标准来看，其CPU运行速度并不快，而只是典型的抗辐射电子设备。这其中的权衡是速度和被一个星际粒子直接击中后还能不断运转的能力。例如，一台典型的PC机里的双核CPU将在太空中无法长时间正常运转（同理，该PC机的其他大部分电子设备也不会正常运转）。

还有一些科学目标，它们驱动着软件的功能和性能需求。所有需求必须满足太空飞行器的计算环境的约束条件。经过无数次的权衡抉择之后，最终产品必须能够运转，而且在飞行任务期间不能出现致命的错误。因为，对于机器人太空飞行器，任何错误都足以导致飞行任务非正常终止，所以，在点燃火箭、飞向蓝天之前，有必要保证所有的方面都是正确的。

在2008年5月25日，凤凰号火星探测器安全降落在火星的北极地区。图3-1是一位艺术家所画的凤凰号着陆效果图。和巡游在火星赤道附近相对温暖的地区的探测器不同，凤凰号探测器是位于贫瘠荒岛上的一个

静止的探测器，荒岛上气候寒冷，大气压力相当于在地球上约10万英尺的高度时的压力。火星上稀薄的大气的主要成分也是二氧化碳。虽然算不上是一个理想的度假胜地，但却是一个寻找古代固态水的好地方。



图　3-1：凤凰号火星探测器在火星上的着陆效果图（图像来源：美国航空航天局和喷气推进实验室NASA/JPL，见彩图10）

该探测器的任务是寻找可以证明水存在的直接证据——可能在火星表面下以冰的形态存在（顺便提一下，探测器找到了冰），以及能够说明火星曾经提供了适合生命的栖息地的证据。由于飞行器的着陆点的位置靠近极地，它的寿命是有限的；当火星的冬季来临时，凤凰号几乎肯定会结束“生命”。在高纬度的着陆点，该探测器在一个完全黑暗的、被固态二氧化碳所覆盖的极寒冷的（-90℃或者更冷）冬天下存活的机会是非常非常渺茫的。

我是凤凰号探测器的成像软件的首席软件工程师。在本章中，我将

分享在设计凤凰号火星探测器的成像飞行软件的各种各样的数据处理策略的过程中所做的一些思考。因为该软件负责处理在火星表面的所有成像相关的任务，又具备“飞行软件”的使命，所以在喷气推进实验室/美国航天局(JL/NASA)的术语中，该成像软件又称为“成像飞行软件”。

对于凤凰号火星着陆探测器，其面临的挑战是同时捕获并处理四个感光元件(carge-coupled device,CCD)成像器的数据（类似于普通的数码相机的功能），而且所有这些操作都是在探测器的主计算机预分配的非常有限的内存中完成。不仅如此，图像可能还需要在传输回地球之前使用一个或几个不同的压缩方法进行压缩。此外，一些最终的数据产品（即图像）需要被细分为不同的分段，每个分段都有自己序列化的数字数据头，这种方式可以满足高效存储在太空飞行器的闪存中，并且当数据包从火星传到地球时，可以减少数据包丢失等原因所造成的数据量的丢失。最终生成的嵌入式代码，在凤凰号火星着陆探测器正常运转的生命周期内，获取并处理了超过25000张的图像。

一些背景

在我们深入了解数据处理之前，最好简单地介绍一下“戏剧的主角”：成像器（也称为照相机）和太空飞行器的计算机。

凤凰号火星着陆探测器的主计算机的CPU是一个RAD6000，其最大时钟率为20MHz，另外它也能够以较低的时钟率来运行，以节省电池能量。这其中没有什么尖端的技术，只是一台防辐射的、把内存和闪存都混合在VME电路板的第一代PowerPC。CPU完成处理着陆任务后，它的主要功能是和地球通信（用专业术语来说，就是“上行和下行”，接下来将会详细介绍），监测太空飞行器的健康状况，通过从地球上发送的各种命令来协调各种科学设备的运行活动。它采用了WindRiver公司的VxWorks实时操作系统(ROS)，以及太空飞行器承包商Lockheed Martin公司提供的各种扩展软件。所有的飞行软件都是用C语言实现的，代码编写遵循一组特定的规范。

“上行”和“下行”

从太空飞行任务的专业角度来看，术语“上行”和“下行”指的是在地面和太空飞行器的控制器之间传送数据或命令。“上行”指的是传送命令或者数据到太空飞行器。“下行”发生在太空飞行器向地面发送回数据。

正如生活中的许多事情，指向飞行器顶部的天线并点击“Push To Talk”（对话）按钮，从来就不是一件很简单的事情。“上行”的数据

或者命令必须首先通过审核，甚至是通过一些模拟来确保一切都准确无误。然后，命令和数据将会传递给命令控制器，当存在“上行”任务时，该控制器将会调度该任务（或者用太空术语来说，即“辐射”(rdiated)）。最后，这些数据进入NASA的Deep Space Network(DSN)通信系统，并从太空中辐射出来。但这并非最后一步，因为对于凤凰号探测器，它还必须通过绕火星轨道飞行的人造卫星进行中转，因为凤凰号探测器本身无法直接和地球通信。当该人造卫星运行到火星水平线时，凤凰号探测器将会监听新的“上行”数据。

“下行”也一样错综复杂。同样地，需要人造卫星作为中继，从凤凰号探测器接收数据，然后把这些数据传送回在地球上的NASA的其中一个DSN接收天线。然后，将会对这些数据进行各种处理和中转操作，直到把它们最终保存在JPL。如果是图像数据，JPL的Mission图像处理实验室将会对这些图像重新组合，提供给Arizona大学的科学运作中心的那些热切期待数据的科学研究团队。

凤凰号火星着陆探测器包含三个主要的照相机来完成表面科学成像：立体表面成像器(Sereo Surface Imager,SSI，有两个CCD感光元件）、机械臂成像器(Rbotic Arm Camera,RAC，只有一个CCD感光元件）、晶体管光学显微镜(MCA Optical Microscope,OM)成像器（同样，有一个和RAC相同的CCD)。图3-2显示了立体表面成像器SSI的飞行模型，图3-3显示了连接到机械臂的机械臂成像器RAC。光学显微镜放置在晶体管装置的内部，它本身看起来像是着陆器甲板表面上的一个黑盒

子。

设计时所面临的挑战是设计一种方式，用以完成以下几个步骤的处理：从每个照相机上下载图像数据；把数据保存在一个预分配的内存空间；对数据进行处理，删除已知的像素点缺陷；修剪或者扩展图像；根据命令行参数进行压缩；然后对图像进行切片和切块(sice-and-dice)，把图像切分成各个数据包，把这些数据包传送给主机的“下行”(dwnlink)管理任务，从而传输回地球。

在立体表面成像器SSI中，每1024×1024像素的CCD感光元件能够生成2MB的数据，或者是100万个12位像素的数据。因为SSI是一台真正的立体照相机，它的成像器通常叫做“眼睛”。而且，SSI看起来确实有点像老式的机器人脑袋。机械臂成像器RAC和晶体管光学显微镜成像器OM每个都包含一个512×256像素的CCD感光元件成像器，每个成像器生成131072（或262144字节）个像素的数据（从现在开始，对于这两台成像器，我都将称之为RAC/OM，因为从成像软件角度出发，它们是完全一样的成像器）。实际上，CCD成像器的每个像素只占用12位的空间；标准内存的一个“字”长是16位。如何处理剩余的4个未使用的位，在设计阶段引发了一些有趣的讨论，这些讨论我将在下章中讨论。SSI、RAC和OM生成的所有图像都是单色的，没有彩色的图像。彩色图片是由在地球上使用过滤器或者特殊的照明技术，获取到独立的图像进行合成而来的。

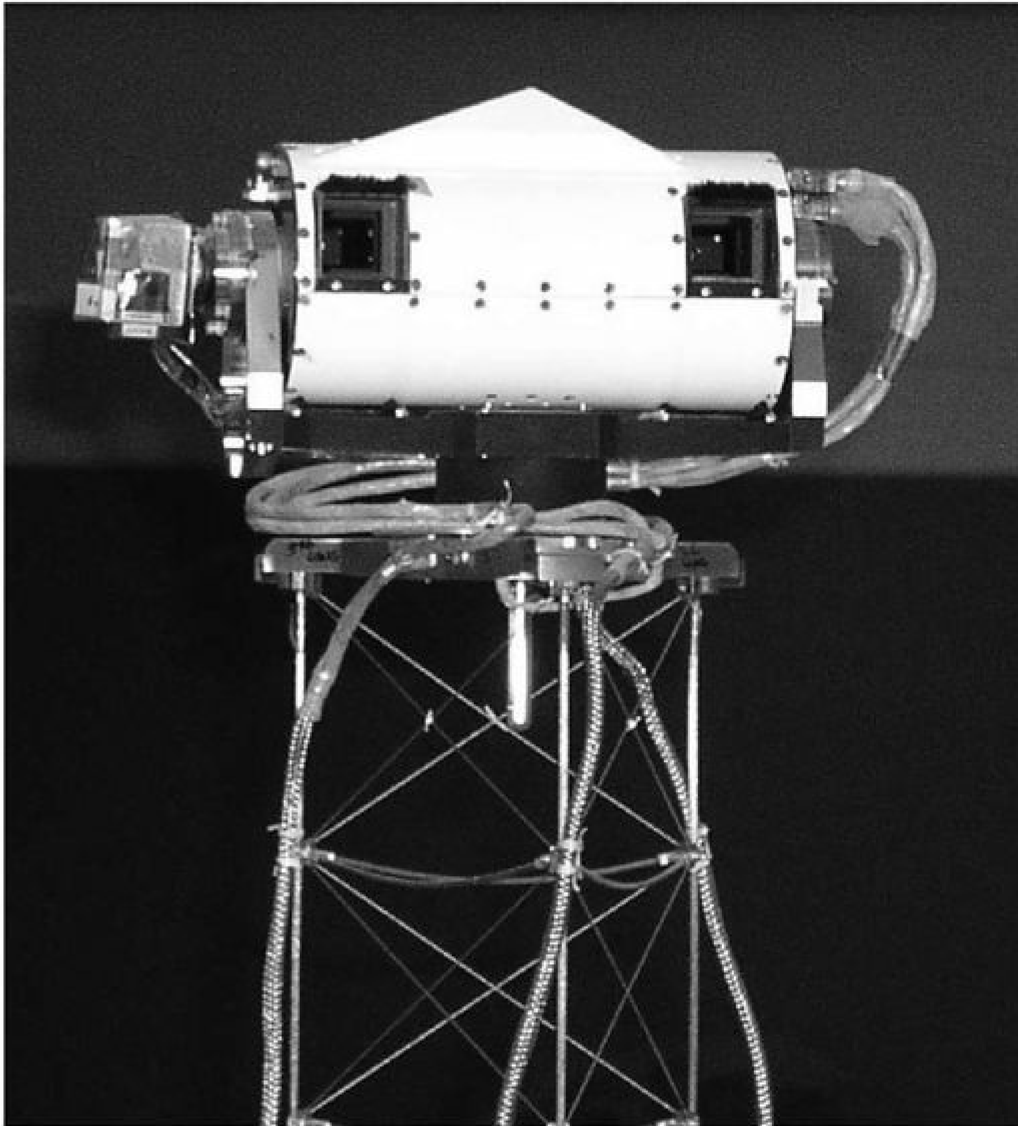


图 3-2：立体表面成像器（图像来源：Arizona大学/NASA/JPL，见彩图11）

必须指出的是，虽然成像软件控制OM CCD来获取图像，但该软件和晶体管(MCA)设备、光学显微镜本身的电子机械控制并没有关系。后二者的控制是由JPL的MECA团队实现的一个独立的实时任务来负责处理的。

虽然以当前消费者的数字照相机的标准来看，凤凰号火星着陆探测

器的100万像素并不算高，但是其使用的CCD成像器是为科学成像而专门定制的。照相机中的每个CCD感光元件设备都花费了数万美元，而且生产制造的数量有限。这些设备可靠、健壮而且精确，而且每台CCD都进行了全面地测试，而且为敏感度、噪音和缺陷等做了像素级别(pxel-by-pixel)的特性定制。正是这种特殊级别的特性化以及CCD生成的参考数据，把科学的CCD设备和消费者使用的普通照相机区别开。精确的特性化使得研究人员对图像数据能够精确地表现照相机所捕获的镜头充满自信。但是它也是影响CCD代价的主要因素。

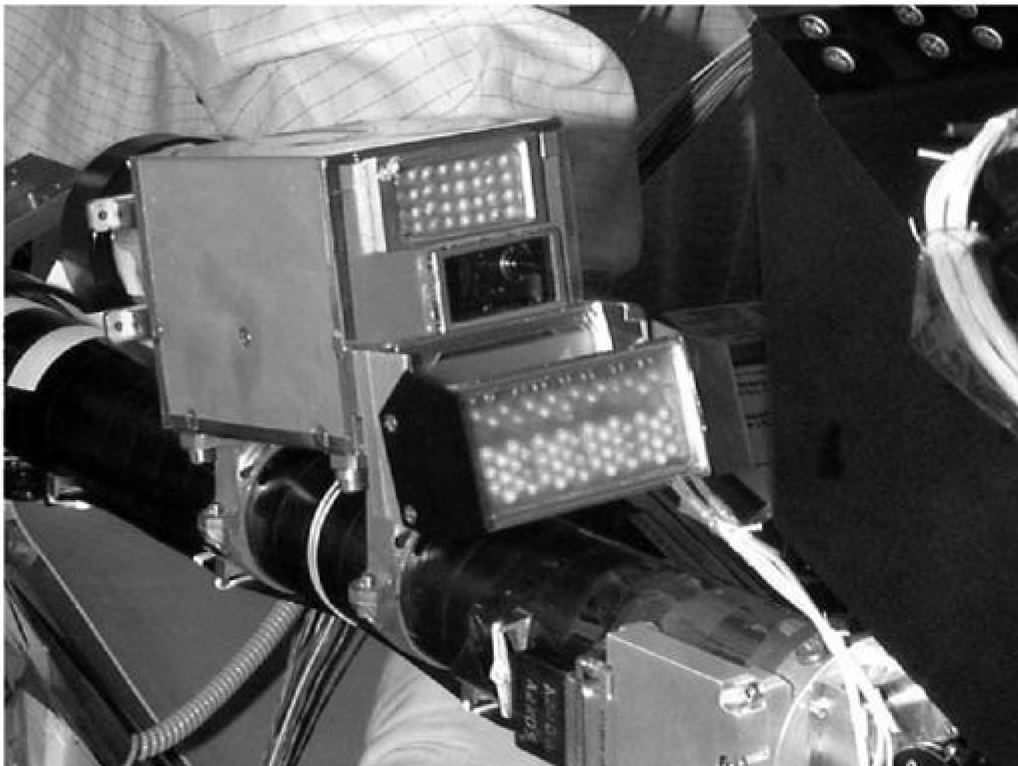


图 3-3：机械臂照相机(Rbotic Arm Camera，图像来源：Arizona大学/Max Planck/NASA/JPL)

数据是否打包

对于任何高度受限的嵌入式系统，其软件需要同时满足操作需求和执行环境的约束条件。正如人们可能预期的那样，这些约束条件往往并非总是互补的，因此在设计实现过程中，需要做出一些权衡决策。SSI和RAC/OM照相机都使用了12位的转换或像素数据，这带来了我们的首次主要的权衡决策：数据打包。对于二进制数据的概述，请参阅后面的“二进制数据”部分。

在任务的早期设计阶段，大家产生了对12位像素数据进行打包的想法，还引发了一些有趣的争论。由于只有有限的内存可用于图像数据存储，对12位像素的数据进行打包存储在16位的内存空间中，这个想法还是很吸引人的。这里的“打包”，指的是对这些12位像素的数据进行连续存储，在这些位中间不“浪费”任何空间——实际上就是忽略16位的内存边界限制。但是更高效的数据存储换来的代价是处理时间变长（解包、移位和重新打包）。一个数字图像就是一个数组（该图像需要作为一维数组处理还是二维数组处理，依赖于需要在它上面完成哪些操作），因此，在图像上的任何操作需要利用对数组的一个或多个算法循环来处理其图像数据。我们期望通过RAD6000处理的数据量非常大，因此，即便是在20MHz的时钟频率下，即计算每个CPU循环时间，该速度也将是慢得让人无法接受。

最后，我们决定“浪费”一些内存空间，把12位的像素存放在16位的

内存空间，使得处理简单，避免占用任何不必要的CPU资源。这个决定也是由以下早期的想法所驱动，即通过就地执行所有的图像处理和压缩操作，避免使用多个大的处理缓存或者结果数组。如果对数据打包，将会使得就地执行这些操作变得异常困难，而且最终代码也将会变得过于复杂，导致整个我们想要实现的就地处理的想法都变得不可行。

二进制数据

数据即信息。在计算机系统中，它是通过数值来表示的，因为计算机CPU所处理的就是数值。数据能够表示文本，在该文本中，每个字符有一个唯一的数值，或者通过对每个像素或者图片元素进行编码来表示图像，在一个图像中，由一个数值表示它的强度、颜色以及这二者特征的组合。给定一个合适的数值编码方案，计算机可以处理一个人可以想象到的任何类型的数据，包括音频、电位、文本、图像甚至是定义猫和狗区别的一组不同特征。但是不论数值表示的是什么数据，对计算机而言，它们都仅仅是数字。我们提供规则来说明该数字应该如何编码、处理、显示和解释。

数据有不同的大小值，该值取决于其表示的是什么。举个例子，一扇窗户或者一个防盗报警系统的门锁只需要1位（或一个二进制数）来表示其可能的两种状态：开或关，即0或1。为了表示一个英语字母或者发音，需要100个左右的数字，每个数字通过8位的数据来表示。现代计算机通过二进制形式工作，因此8位可以表示从0～255的任何一个数字（如$11001000_2=200_{10}$）。在很多计算机中，有更合适的形式来表示二进

制数值，如典型的八进制或者十六进制。对于16位或32位的CPU，正如凤凰号火星着陆探测器所采用的，内存可以通过16位的字长方式来高效访问。想要访问低于16位（如8位）的字长方式，其效率实际上可能会很低，因此对于大于8位而又小于16位的数据，通常都是以16位字长的内存方式来存储，有几位没有使用。这正是凤凰号火星着陆探测器所处理数据的方式，因为CCD成像器的电子设备每个像素生成的是12位的数据，而飞行器的内存是通过16位或32位的存储方式组织的。

三个任务

对于凤凰号火星着陆探测器所采用的VxWorks实时操作系统环境，实际上确实没有什么方面是和普通的计算机用户可能会想到的某个项目相同。不在磁盘上加载任何东西（不存在磁盘），而且当系统初次启动时，计算机所做的每件事都被加载到内存中。该实时操作系统实际上只是一个大程序，包含很多几乎同时在运行的较小的子程序。这些较小的子程序又称为任务或线程，基于以下因素来执行：资源的可达性，如时间事件或者I/O（输入/输出）设备；这些子程序在更大范围中被分配的优先级（优先级高的任务比优先级低的任务能够获取更多的机会运行）。

一开始就显而易见，对于表面图像处理，至少需要完成两项任务，每台照相机完成一项任务。SSI成像器和RAC/OM成像器有很大区别，它们包含不同的命令集和操作特征。SSI使用所有新的控制器硬件，并且结合了和那些用于Mars Rovers（译注：火星巡游器）完全一样的CCD成像器。RAC和OM成像器最初是由德国的Max Planck研究所建立的，而且已经应用了一段时间（其中一个最初设计应用于Huygens探测器，在Titan着陆）。RAC/OM控制器硬件实际上是命运多舛的Mars'98任务中的飞行器备用设备，当其飞行器引擎在火星表面上方几百英尺处不正常终止时，该控制器也结束了其“悲惨的命运”。但是从每个照相机获取的数据仍然需要处理、压缩，然后“下行”（向地面传输），而且这些操

作并不依赖于物理数据源。图像数据的每个像素都是12位，这些数据中真正有区别的是几何信息（高度和宽度），以及最终需要处理的图像数据量。

虽然太空飞行器集成团队(Lckheed Mar-tin)期望保持科学设备的任务数量是最少的，在早期就显而易见，在两个照相机任务中都重复相同的图片压缩和“下行”功能并没有多大意义。由于程序的存储空间有限，这种重复操作会浪费资源；因此，对代码做必要的修改，然后验证这些修改可以达到事半功倍。最后，我们达成的一致意见是使用三个任务：一个任务处理SSI，另一个任务处理RAC和OM照相机（这两个任务不能同时执行，因为接口硬件不允许它们同时执行），第三个任务将作为共享资源，利用由两台照相机的控制任务所生成的数据，执行图像压缩和“下行”处理，因而它是异步运行的。因此，虽然SSI和RAC/OM任务通过控制电子设备来获取图像数据、控制照相机的内部温度、执行运动控制，第三个任务仅仅完成图像数据处理和“下行”操作。

图像处理任务被称为ICS,ICS代表图像压缩子系统(Iage Compression Sub-System)，虽然该子系统最终不仅仅只是压缩图像数据。图3-4中显示了三个任务的流程框图以及它们之间的通信。

使用三个任务的决策确实增加了系统的复杂性，但它也减少了需要的程序存储空间。作为额外的补充，有第三个独立的任务使得改变一个特定的ICS处理功能变简单得多，然后通过使用真正的照相机硬件，对从任何一个成像任务获取的数据或者是采用模拟的图像数据源，对这些

功能进行测试。这些测试，对于后期对项目做更多的扩展压缩测试是极有好处的；当超过15000个的测试图像通过ICS被连续处理，同时使用自动化测试验证其操作。

ICS也包含两种资源模块，它包含静态内存管理（也称为“槽管理”）和图像处理（取样、子图定位、像素缺陷纠正等）。这些实际上并非ICS任务的一部分，而是作为线程安全的伪程序库来支持两个照相机控制任务以及ICS任务。实际情况是由于该设备软件所面临的架构约束，把这块代码放在任何其他地方都不合适，因此，最终把它放在ICS。

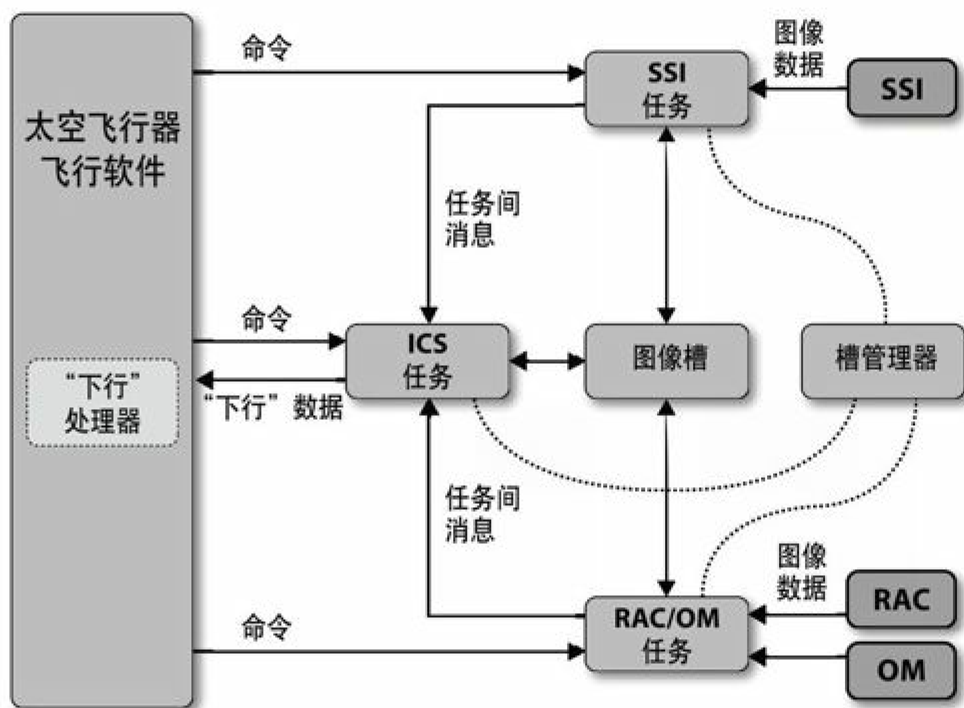


图　3-4：图像飞行软件任务

对图像切槽

本章之前提到过，每个设备任务可以分配到的内存数量是有限的，但究竟有限到什么程度，可能说出来会让有些人大吃一惊，因为现在对于一台桌面台式机，有500M或者1G的内存已经很普遍了。分配给SSI和RAC/OM用于图像数据存储任务的初始内存差230K是10M（确切地说是10255360个字节）。大家讨论过，在太空飞行器着陆后，可以增加其内存分配。这意味着任何内存管理机制都必须很灵活，而这也是设计的基本要求。默认的存储机制需要至少处理四个SSI图片（或者两对，每个“眼睛”包含一个图片），而且至少四张RAC/OM图片，这些图片都存储在相同的内存空间中。这种零散的大小分配方式意味着至少在初始阶段，不可能“挤进”超过四张的全尺寸大小的SSI图像。

在嵌入式系统中，使用动态内存分配通常被认为是一个非常糟糕的想法。为了避免内存碎片、内存泄露、空指针、莫名其妙的程序崩溃以及可能完全无法完成任务，飞行软件编码规范禁止使用动态内存分配（C的malloc函数及其家族）。这意味着不论分配给成像软件的内存有多小，它都需要自己在该内存范围内管理图像数据，而且软件必须健壮、可靠。

为了满足以上需求，成像软件的解决方案是在启动时，使用一组函数作为内存管理器，用来为ICS预分配内存。内存管理器是图像数据处理的核心组件。为了避免冲突，采用阻塞信号来控制三个成像任务间的

共享访问（实际上，在太空飞行器软件中的任何一个任务都可能使用共享内存，但实际情况是只有照相机使用共享内存）。

静态内存分配被划分成“槽”，它可能大到足以容纳一个全尺寸大小的SSI图像，或者是较小的RAC/OM图像。图3-5显示了ICS图像存储空间的默认组织方式。

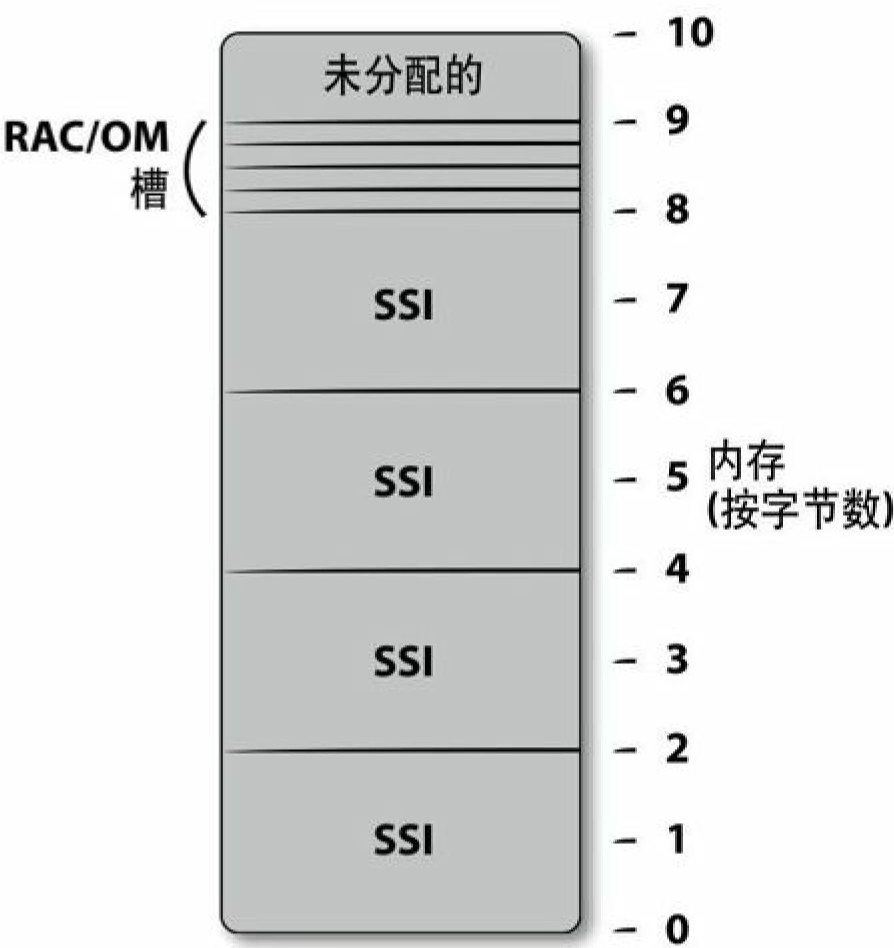


图 3-5：默认的图像“槽”分配

这只是一种可能的配置方式，每种类型的槽的个数可以通过接收从地面传输上来的命令而随时改变。

每个图像还包含一个包含头数据的关联结构。图像头数据记录了以下信息，如定义生成图像的照相机的代码、曝光时间、选择的图像处理选项、图像维度、可能使用的光学过滤器以及图像如何压缩（如果采用了压缩方法）。部分的头数据是由生成图像的设备任务生成的，其余部分是由ICS在把图像数据发送给太空处理器“下行”的处理程序之前生成的。由于头数据并非图像数据本身，它存储在一个独立的内存槽，直到需要执行“下行”操作时才使用。每个图像槽和其关联的头数据需要通过协力的方式进行跟踪和处理。

如图3-6所示，内存管理器实际上只是一组函数集合，这些函数在一个结构数组上执行操作。内存槽的当前状态是保存在数组中，这些数组实际上构成了一个物理内存空间及其内容的动态模型。

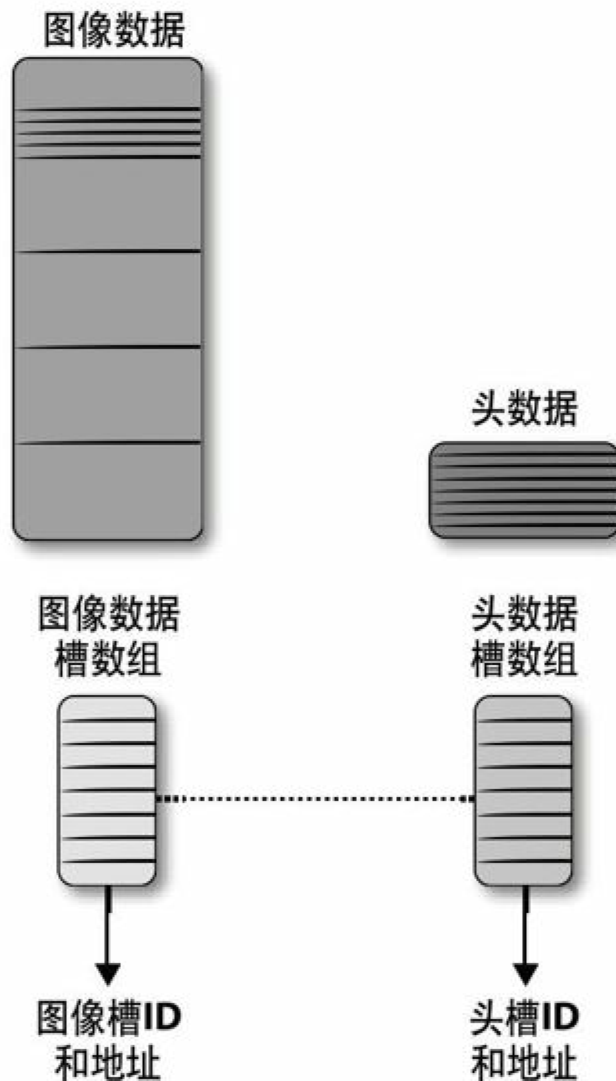


图　3-6：内存槽管理数组

其中一个数组包含图像数据的结构体，每个图像槽对应一个结构体。该结构体通过C语言类型定义如下：

```
typedef struct{
uint16_t slot_status;   /**<Owned or unowned  */
int16_t slot_owner;   /**<-1 if slot is unowned  */
int16_t slot_size;   /**<either RAC/OM or SSI sized  */
uint16_t*slot_address;   /**<address of data space of slot  */
}ics_img_slot_entry_t;
```

第二个数组包含指向头数据入口的结构体，其定义如下：

```
typedef struct{
uint16_t slot_status;   /**<Owned or unowned  */
int16_t slot_owner;   /**<-1 if unowned  */
int16_t img_id;   /**<associated image data slot number  */
uint16_t hdr_data[ICS_HDR_SLOT_SZ]; /**<array for header data  */
}ics_hdr_slot_entry_t;
```

注意：图像的头结构体包含该图像ID的入口。这一点是很有必要的，因为槽可以以任意次序进行分配和释放，而且不能保证一个图像槽入口的索引和它原来关联的头数据槽的索引是同一个索引。我们采用图像ID把图像和头数据入口绑定在一起，而不是依赖于数组中的索引偏移来保证图像和头数据入口是同步的。

由于可以动态重新配置图像的槽分配，内存管理器可以适应于特定的任务。一方面，如果为火星上的某天制定的计划（或者称一个Sol)和通过SSI成像相关，那么我们就可以通过槽配置来最小化RAC/MECA的大小分配，即设置为默认的配置。另一方面，如果该计划和很多的RAC或OM图像相关，那么我们可以对内存进行配置，使得它能够处理上限39个、较小尺寸的图像。

传递图像：三个任务间的通信

一台照相机的图像数据刷新会通过其中一个照相机任务写入槽中。在执行完任何必要的像素纠正和子图定位操作之后，该照相机任务会通知ICS有新的图像可以处理。然后ICS根据输入的命令，对该槽中的图像执行就地压缩（不论是有损的还是无损的），然后对图像数据进行打包，传递给“下行”处理程序。只有完成了“下行”处理，该槽才会被释放，供新的图像使用。从图像曝光到为SSI照相机传送图像的事件顺序如图3-7所示。

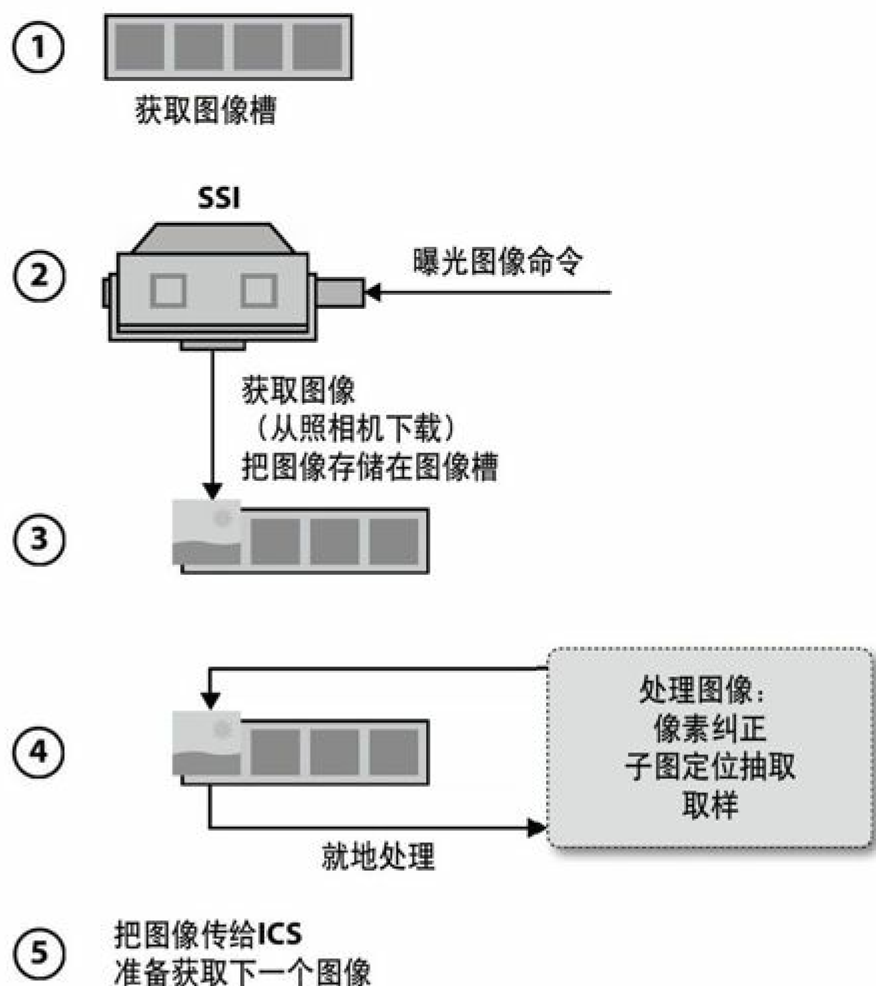


图　3-7：图像获取和传送序列

如图3-6所示，整个事件的次序是根据图像槽的可达性而定的。如果无法获取槽，照相机任务将会等待一段可配置的时间，使得ICS可以完成压缩和图像“下行”处理，这样就有一个图像槽变得可用。如果ICS在那段时间内不释放槽，该照相机任务就会给地面的操作员生成一条错误信息，直接把该图像保存在照相机中（因为实在没有其他地方可以保存该图像）。

一旦一个照相机设备任务获取一个槽，它在把该槽交给ICS之前，

就“拥有”该槽，而交给ICS之后，ICS就拥有该槽。拥有权的验证是基于槽管理器在初始化该槽时所分配给它的槽ID（即ID数值）、图像ID代码和照相机设备任务ID。当某个任务把一个槽交给ICS时，ICS会验证这些ID和之前记录下来的该图像槽的ID是否匹配。

正如之前所述，ICS是异步方式运行，且和任意一个照相机任务并没有密切关系。可以通过利用VxWorks的内置消息队列系统，并且在内存槽管理器中使用共享函数，把ICS和照相机关联起来。图3-8显示了这些步骤的消息序列图(MC)表示方式，这些步骤包括从照相机获取数据到最终把它们传输给“下行”程序。

使用内部消息队列（S/C FSW操作系统管道，代表太空飞行器的飞行软件），这将允许任意一个照相机任务使用和执行从地面“上行”的命令相同的机制，并向ICS发布命令。该命令会保存在队列中，直到ICS完成了其当前的任务。只要有可用的槽来存储图像数据，照相机就会继续获取图片，ICS将轮流检索和处理这些数据，直到队列为空，而不需要考虑照相机当前的工作状态。

注意：图3-8没有显示在成像过程中的错误检查。总之，错误检查和出错处理的代码行数大概和实际上执行处理或者处理该数据的代码的行数相同。在设计生命周期的早期就采用了失效模型效果关键性分析(Filure Mode,Effects,and Criticality Analysis,FMECA)技术，该技术在软件实现和出错处理能力上具有指导作用。

ICS对数据流进行序列化，但是使用图像槽和命令消息队列允许一组

图像可以连续快速地（相对而言）获取。这也意味着对于图像获取，存在一定的时间间隙(tming margin)，这减少了由于等待图像“下行”处理，操作被挂起的几率。早期的测试命令序列证明了可以执行如下操作：制作短篇“电影”（这么说是因为从SSI照相机下载一张图片大约需要6秒），或者使用不同焦距的RAC或者OM生成大的（30多张图像）数据集合。

获取图片：图像下载和处理

从图像曝光到最终传送给ICS，这期间发生了很多事情。系统中的每个照相机都有自己的控制电子设备来处理命令和把从CCD获取的模拟信号转换为12位的数字值，然后把数据存储在硬件缓冲区中，直到飞行软件可以把数据下载到图像槽。

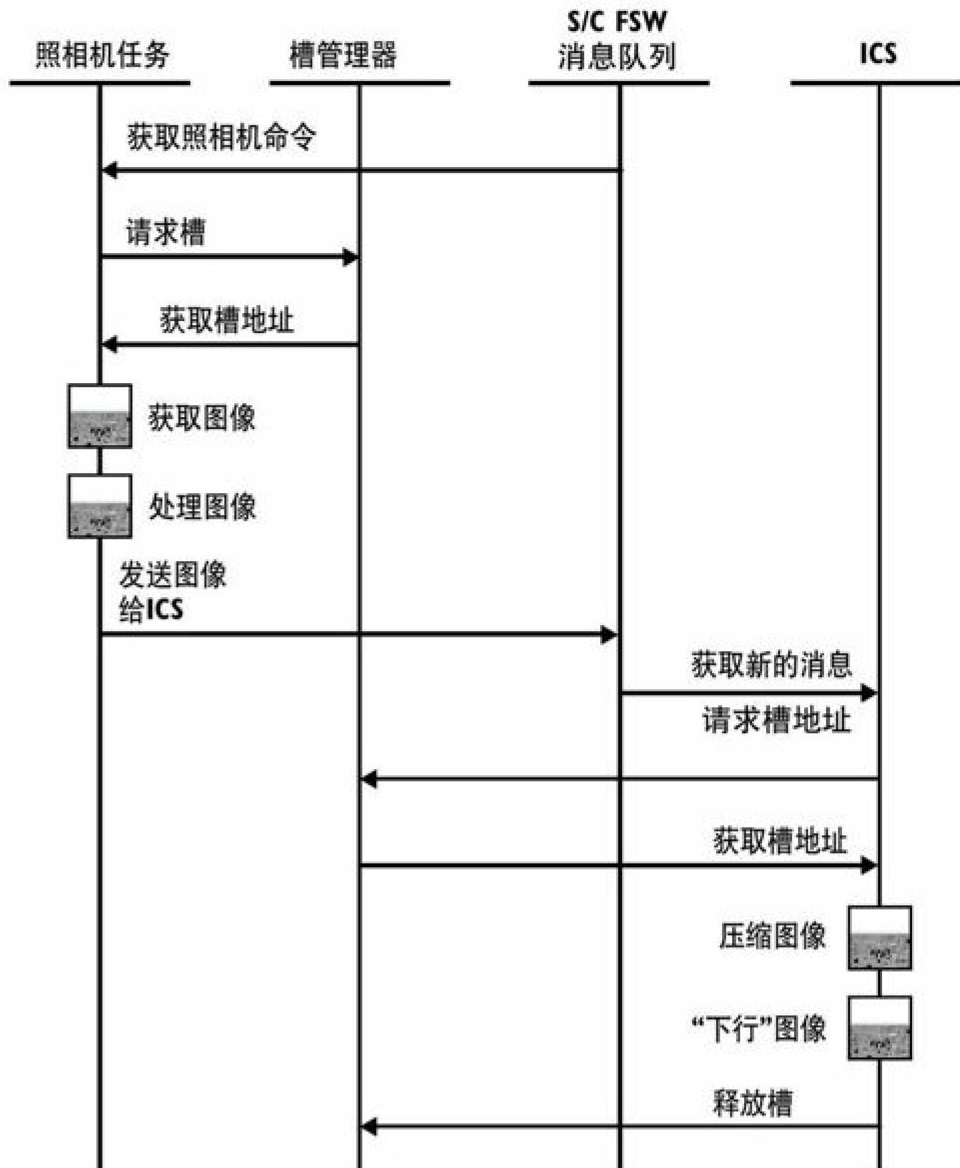


图 3-8：图像获取、处理和“下行”处理的MSC表示

以上所有这些都是在防辐射的可编程门阵列设备中的嵌入式逻辑控制之下发生的。

照相机一旦获取到一张图像，并把它写入图像槽，就会对该图像执行各种方式的处理，每种处理都是在图像槽的约束下就地执行。由于没有额外大的（图像尺寸）缓冲区可以供处理过程或者保存结果数据使

用，因此，只有几个小的缓冲区来保存必要的中间结果。使用就地处理方式是设计成像软件的关键因素，它允许三个任务在整个系统中只占用很少的内存。图3-9显示了对于凤凰号火星着陆探测器成像飞行软件，多缓冲区的处理方式和单缓冲区（即槽）的就地设计策略之间的区别。

我们采取就地单缓冲区处理方式，是为了满足图像处理需求，并且占用的内存仍然在预分配给照相机的内存范围之内，在早期设计时采用的一个权衡。虽然这种设计处理方式，确实满足内存需求，其不足之处是无法提供“undo”操作。如图3-10所示，如果在图像处理时出现错误，那么会导致或者整个图像丢失，或者是返回的是部分错乱的图像。

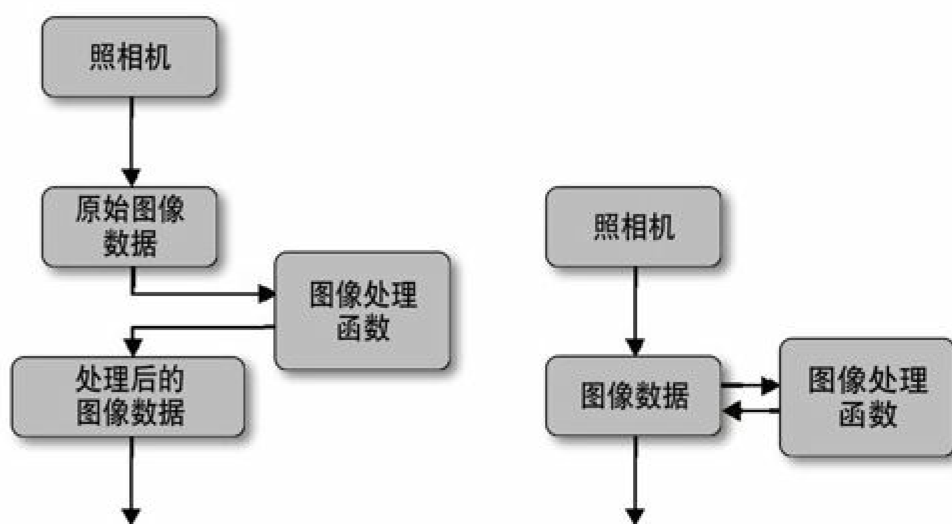


图　3-9：多数据缓冲区和单数据缓冲区

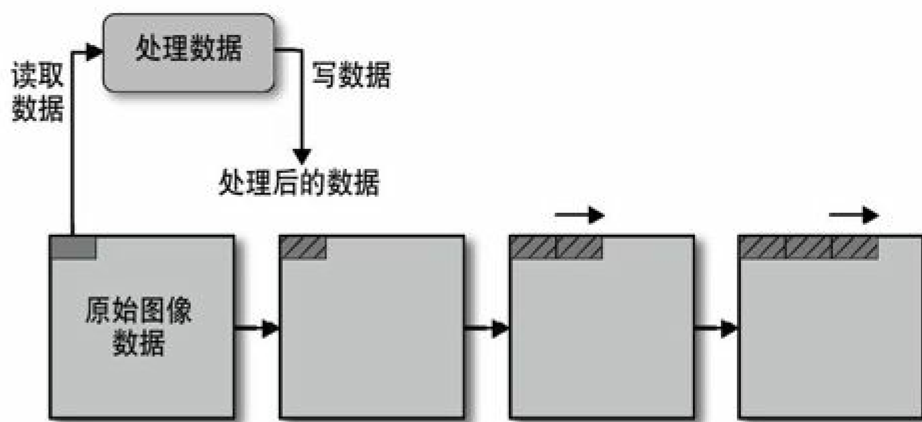


图　3-10：就地数据处理

就地数据处理方式的处理算法是应用于图像槽中的全程数据处理，包括读数据、处理数据及写回结果数据。有些算法，如像素纠正算法，并没有改变图像的几何信息，而只是基于已知一些“坏”像素的一个“上行”表，简单地对单个像素值进行修改（一个像素有可能是“坏”像素有两个原因：一是其没有相邻的像素那么敏感，二是它可能过于敏感）。在空间环境中，人们认为奇特的宇宙射线可能会穿透CCD上的一个像素，导致该像素有缺陷。其他的操作，比如图像子图定位，从原始图像中抽取一块区域，把该区域写回到槽中，然后相应地调整高度和宽度参数。抽样技术采用了一种数学均值计算的方式，通过把像素按4、9或16分组的方式，每个组生成的是单一的结果像素，从而减少图像大小。结果像素在图像大小上相应地缩小了1/2、1/3或者1/4；而最小化“梯度步长”的处理方式，通常是在当图像通过子抽样技术，只保留每第2、第3或者第4个像素，而抛弃其他像素来达到减小图像大小。这种操作也把修改后的结果数据写回到图像槽中，并相应地调整了其几何参数。

当一个成像任务完成了根据命令执行的处理操作，它就会向ICS发送一条消息（如之前所述），然后执行消息队列中的下一条命令。如果有可用的图像槽，该任务会执行获取另一个图像的处理操作。

图像压缩

正如由机器人完成的任务生成的数据非常宝贵，需要返回这些数据的通信带宽也是非常宝贵的。对于较小的图像，比如那些通过子图定位或者抽样操作，图片大小已经减少了，因此直接执行“下行”操作而不做压缩处理是可行的。更大的图像，比如全尺寸大小的SSI图像，“下行”操作会消耗很多带宽，因此在这种情况下，通常采用压缩方法来解决。

ICS采用像素映射和扩展，提供了两种压缩和减少图像大小的方式。对于某个特定的图片，采用哪种压缩或减少图像大小方式，主要依赖于图像需要达到的保真程度，高保真被认为是图像的一个必要方面。在一些情况下，每个像素8位就足够了；而在其他一些情况下，JPEG压缩本身造成的图像保真损失是可以接受的；而对于一些情况，图像需要保持尽可能高的保真，则可以采用无损压缩的方式。

在ICS内部，一台JPEG压缩器采用所有的整数算术计算和就地操作，提供所谓的“有损”压缩方式。JPEG被认为是有损的，因为其压缩过程丢失了部分图像数据。JPEG可以通过命令，对图像数据实现不同程度的压缩。最终代码是松散式地基于Mars'98使命的JPEG压缩器；虽然凤凰号火星着陆探测器的ICS的实现只采用了其部分原始代码。原始的JPEG压缩器使用的是浮点数乘以全尺寸大小的图像数组作为缓存，并采用动态内存分配方式。对于这种方式如何在飞行软件上正常工作，我

仍然感到很困惑，不过它确实能够正常工作。在压缩代码中使用浮点数来表示像素数据，这也意味着对于每个图像，比起16位整数的原始图像表示方式，浮点数占用了其四倍的内存空间。

第二种压缩方式，也称为Rice无损压缩(Rce Lossless)或者Rice压缩，采用了由Jet Propulsion实验室的Robert Rice开发的一种算法。该Rice算法可以对图像数据实现几乎2：1的压缩效果，且没有数据损失。而JPEG算法在压缩过程中丢失了部分数据。Rice压缩方法也是在图像槽中就地对图像进行压缩。

两种无压缩的缩小图像大小技术或者采用查询表，把12位的像素值映射到8位的像素值，或者采用位缩小技术，对像素数据向右移动4位，生成一个每个像素8位的图像。JPEG和Rice压缩函数都接受12位或者8位的图像数据。

是否采用有损的JPEG压缩方式，通常需要对各种因素进行权衡，比如数据需要达到的精确度、可用的带宽大小、太空飞行器的主机有多少可用的“下行”存储空间、有多少时间可用于执行压缩（之前提过，RAD6000的最高速度是20MHz，因此压缩1M像素的图像数据可能需要超过1秒钟）。

当使用JPEG压缩方式时，采用的压缩比率是通过命令行参数决定，该命令行参数指定结果数据在最差情况下的压缩比率。换句话说，ICS不是指定一个“质量”因子（该因子通常表示JPEG压缩方法需要达到的压缩级别），而是使用一个可扩展因子，自动计算得出需要的压缩级别。

这是基于全局“图像熵”(iage entropy)的“快查”(qick-look)分析。图像熵是图像“繁忙”程度的估计值，熵值越高的图像（在亮度上有很多细节和变化，如有很多鹅卵石的地面，阴影对比分明）将需要更高的压缩设置来满足最终的目标图像大小限制。对于熵值低的图像，比如飘着一些云彩的Martian天空，没有太多的细节和变化，因而只要较少的压缩量就能满足目标图像大小限制。

JPEG压缩算法扩展因子也用于把原始图像划分成不同的分段。这些分段一次一个地传给JPEG压缩器，其输出被写回到图像的槽。在执行“下行”操作之前，图像槽的最终结果是一组小的、自包含的JPEG图像，这些图像的全部大小不大于由命令行指定压缩比率的原始图像大小。

Rice压缩器包含它自己的嵌入式分段方法，而且其“下行”操作是简单地通过读出压缩数据，这些压缩数据通过小的分包方式，分包大小适合太空飞行器主机的闪存大小。查询表和减少位的方法的输出也是简单地读出闪存大小的包，用于“下行”操作。

“下行”或一切都从这里向下传输

太空飞行器的飞行软件处理过程的最后一步是传送图像给“下行”管理器。一些科学设备可以简单地把数据传给“下行”程序，完成处理；但是，由于海量的数据以及采用分包的方式，结果是ICS本身需要完成很多的预处理操作。

对于JPEG数据，这意味着需要分别单独处理每次的图像压缩分段。一个序列的第一个和最后一个分段通常包含一个全尺寸大小的图像数据头。中间分段的头数据大小较小，包含一个图像ID号和序列号。由于每个分段都是从图像槽读取数据，所以采用了头数据。采用滑动窗口的方式向外读取数据，使得分段在以闪存大小分包组合时，可以对分段执行端对端(ed-to-end)地打包。反过来，它允许“下行”处理器最大化使用临时闪存存储空间，因为如果一部分图像对应的分段的熵值很小，那么一些压缩的分段可能比其他的分段小。实际上，压缩的分段大小区别很大是很常见的，因此对这些分段执行端对端地打包避免了浪费飞船上任何的闪存资源。

因为数据是由唯一识别的分段组成的，“下行”过程丢失一个分包并不会导致整个图像毫无用处。在JPL，地面上的重构和解压缩软件能够识别出丢失了哪个分段，然后简单地把丢失的那部分图像以全0像素数据来填充。如果后期获取到丢失的那部分数据（这是可能的，因为考虑到数据的“下行”过程是非常曲折的），那么这部分数据就可以用来填充

之前丢失的那部分像素数据。

一旦发送数据给“下行”处理器，ICS就完成任务，并且释放该图像数据的插槽。这整个过程（从图像曝光到完成发送给“下行”程序）需要3～10分钟，这依赖于CPU速度和其他额外必定发生的成像处理，如自动曝光和寻找光线（这两者本身就非常复杂，所以我们不在这里讨论）。

结束语

该设备软件所完成的任务要远远多于照相和处理图像数据。它也管理SSI的三（自由）度运动控制，以及RAC中的聚焦和视角。RAC还支持多色的LED，包括红、蓝和绿，照亮任何可能在机械臂范围内的视角，然后生成彩色图片。SSI和RAC都结合了活动的热控制，这或者是通过使用特定的加热器，或者是故意拖延一个步进电机来达到自我加热。所有这些操作都包含错误检查和容错恢复。总之，该设备软件任务非常“繁忙”。如果让我重新设计该软件，我希望改变的主要方面是，照相机使用的是它们自己的嵌入式处理器，而不是依赖于太空飞行器的CPU。这样会使所有的操作都变简单得多。除了这点，我总觉得对于每个设备任务，“挤进”了太多的操作。换句话说，热控制对于每个照相机本应该是一个独立的任务。这样就可以极大地减少每个任务的复杂度，虽然其代价是增加全局任务间通信的复杂度。然而，在刚开始时，没有足够的理由来这么做，因此当有一些新的加热需求需要对设计做出调整时，该设计已经很固定了（没有冻结，只是非常不灵活）。

最后，对于选择“心跳”检查方式，我真的是很不支持。我原先不知道命令消息队列是为了进行“心跳”检查。这种“心跳”检查方式对设备任务提出了需求，要求每个任务能够放弃任何它们正在做的事，通过“ping”消息定时检查命令消息队列。我相信更好的方式应该是让设备注册一个太空飞行器飞行软件回调函数，该函数可通过异步方式，检查

连续更新的计数器变量的值。如果在一定时间后，该值没有更新，则该设备任务很可能已经崩溃。当时确实对这种实现方式有很多争议，最后采用ping消息的方式，仅仅是因为这是一种通用的方式，而且已有的测试系统就是这么设计处理的。因此，即使系统不是为了需要花费好几分钟来处理的海量图像数据而设计的，但是该设计方式也不会发生改变。

凤凰号火星着陆探测器的SSI和RAC/OM成像软件在设计、实现和测试上有很多工作要做，而且在最后，它完成了在其使命的整个生命周期中应该完成的任务。图3-11是该太空飞行器在火星上的第一天，即Sol 1，从SSI返回的第一批图像之一(S001EDN896308958_10D28R1M1）。

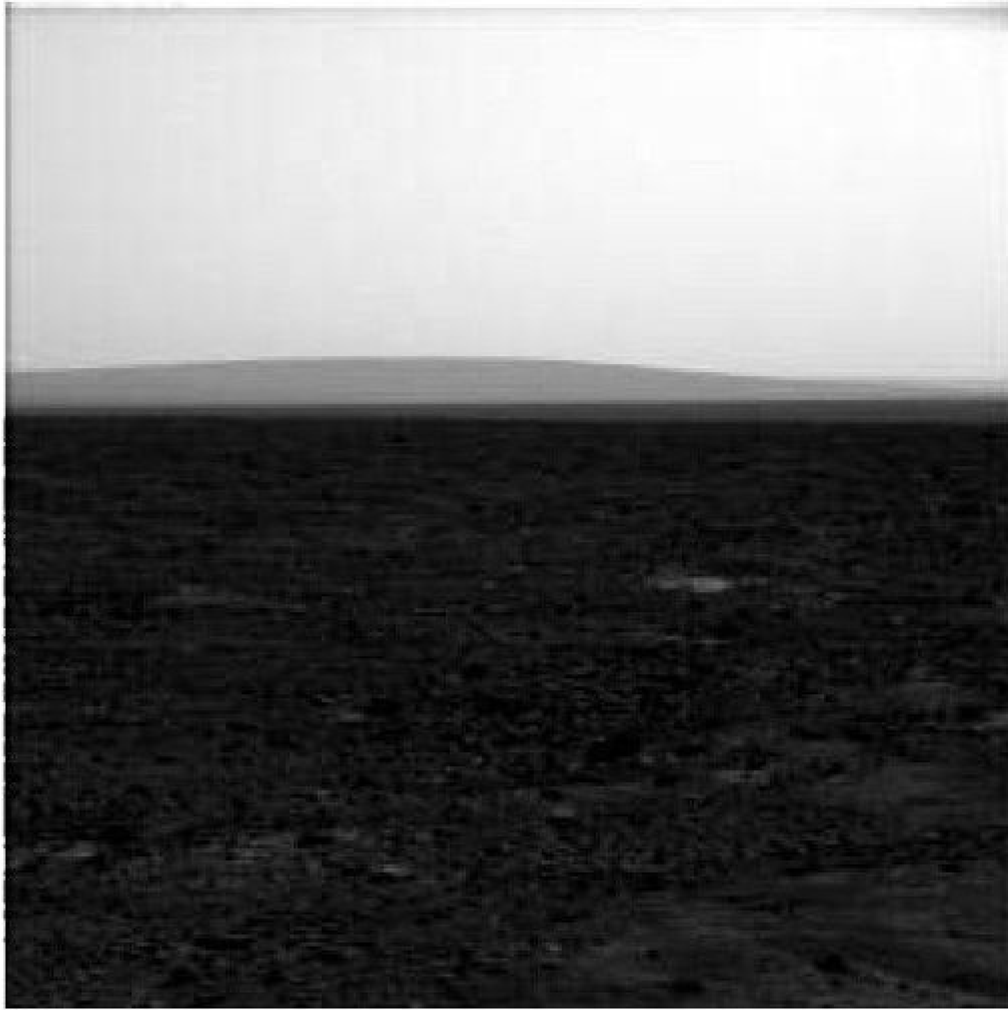


图　3-11：从Sol 1的SSI返回的图像（图像来源：NASA/JPL/Arizona大学）

更多关于凤凰号火星着陆探测器

如果你想对凤凰号火星着陆探测器的使命了解更多，可以查看以下几个主要来源：

Arizona大学的Phoenix网站：·http：//phoenix.lpl.arizona.edu

Jet Propulsion Laboratory(JPL实验室）的Phoenix网站：

·http：//www.jpl.nasa.gov/news/phoenix/main.php

NASA的Phoenix网站：

· http：//www.nasa.gov/mission_pages/phoenix/main/index.html

在JPL,MIPL为各种任务做了很多图像处理，你可以通过以下站点了解更多：

JPL的Mission图像处理实验室： · http：//www-mipl.jpl.nasa.gov/

如果你想对RAD6000 CPU、图像处理或嵌入式系统了解更多，可以查阅Wikipedia:http：//www.wikipedia.org。

第4章　PNUTShell中的云存储设计

Brian F. Cooper、Raghu Ramakrishnan和Utkarsh Srivastava

简介

雅虎公司运营着世界上最受欢迎的多个网站，每月的访问用户有5亿之多。这些Web站点的后台数据系统存储了种类庞杂的数据，如用户信息、照片、餐馆点评、博客博文以及大量其他数据。雅虎开发部署了多套成熟、稳定的数据库体系来支持它的站点，同时这些架构也为用户提供低延迟的数据访问方式来快速加载页面。

但是，这些系统存在着一些严重的局限性。首先，系统扩容困难。为了进行系统扩容，通常需要数月的时间来规划和重组数据，并且在扩容过程中，相关应用的服务质量会受到影响。此外，有些系统可支持的扩展性具有上限值，即使增加硬件支持也无济于事。其次，很多早期设计的系统，其设计思想就是基于单一的数据中心。雅虎已经发展为全球性的品牌，其广泛的基础用户遍及全世界。为了提供良好的用户体验，系统需要复制数据到离用户近的数据中心，这样当用户访问时，页面就可以快速加载。由于数据库系统本身并没有将全球性数据复制作为内建的功能提供，每个应用需要自己构建数据复制，这导致复杂的应用逻辑和薄弱的数据基础。部署一个大规模的、在不同地理空间上都存在数据副本的数据库架构需要很大投入，因此，在这样的架构上，很难快速地开发新的应用或在已有的应用上增加新的特性。

PNUTS系统致力于支持雅虎的Web站点和应用平台，并解决上述局限(Coper等2008）。在设计上，它通过云存储方式，可以有效地处理租

户应用(tnant applications)中复杂的读写工作，并且支持本地全球数据复制。和很多其他的分布式系统类似，PNUTS通过水平数据分区的方式，将数据分布到一组存储服务器阵列上，从而实现高性能和可扩展性。该系统的核心不是复杂的分析和决策支持工作，而是从最初设计就把两个属性作为首要特征：

向外扩展(sale-out)

系统通过分区将数据分布式存储到多台服务器上，扩充系统容量犹如增加新的服务器一样容易，且系统可以平滑地将负载分配到新的服务器上。

地理空间上的数据复制(go-replication)

在PNUTS系统中，数据可以自动地复制到全世界。一旦开发者告诉系统在哪个区域[1](clos，即数据中心）复制数据，系统本身就会自动地完成诸多细节操作，包括故障处理（如机器、链接甚至整个区域）的一些细节操作。

PNUTS系统还设置了一些其他的目标，尤其是，我们希望应用开发者能够专注于应用逻辑本身，而不是操作数据库的具体细节。因此该系统对数据库采取托管方式，并为开发者提供一个简洁易用的API来存储和访问这些数据，而不需要他们自己对一大堆参数进行调优。由于系统采取托管方式，在功能实现上，它能够尽可能达到自我维护。

尽管这些目标都很重要，而且构建一个能够向外扩展且易于全球性数据复制的数据库系统是对公司最有吸引力和直接价值的。随着开始系

统设计，我们清楚地发现很多长期普及和使用的数据库系统机制需要重新考虑(Rmakrishnan和Gehrke 2002）。

实现向外扩展和地理空间上数据复制的核心思想是只进行简单廉价的同步操作，而在后台异步式地实现所有代价高的操作。举个例子，当一个加州的用户想用一个关键字来标注一张照片，她必定不愿意等待该系统把该标签提交到在新加坡的标签数据库副本（从加州到新加坡的网络延迟可能高达1秒）。然而，她还是希望她的新加坡朋友能够看见该标签，因此需要在后台对新加坡的数据库副本完成快速（几秒内）、可靠的异步更新。

另一个如何发挥异步作用的例子是诸如聚集查询和连接操作，它们通常需要在多台服务器上执行数据操作。随着系统规模扩大、负荷加重，这些服务器出现响应变慢或者崩溃的概率也随之增加，因此会增加请求延迟。为了解决该问题，我们可以维护实体化视图(mterialized views)来重组这些基础数据，这样一些预先设定的复杂查询可以通过只访问一台服务器来获得响应。和数据库副本相似，若同步更新每个视图，在执行写操作时非常缓慢，因此我们的策略是采取异步方式更新这些视图。

本章的剩余部分会深入地剖析重点研究向外扩展和地理空间上的数据复制这两大首要特征的涵义。我们将通过一个例子来阐明主要问题、说明基本方法、讨论一些问题及扩展；然后，将PNUTS和其他方法作比较。本章的讨论重点是设计理念，而不是系统架构或实现细节，并且为

了在全局方案中突出某些选择，本章涵盖了一些当前系统版本不包含的特性。

[1]区域设施，或者称数据中心。雅虎在全世界运营着大量的数据中心。

更新数据

用户和Web站点的交互操作会导致数据库的不断更新。因此，我们要解决的首要挑战是为每个更新提供良好的性能和操作一致性的前提下，如何支持大量的更新操作。

面临挑战

假设我们要构建一个社交网络站点，系统中的每个用户都有一条用户信息记录，该记录包含用户名、爱好等。用户Alice可能在全世界都有朋友，她的朋友想看她的个人信息，这些读取请求操作有严格的低延迟要求。为此，系统必须保证Alice的用户信息记录（其他人的也类似）在全球都有副本，这样她的朋友们就可以访问该记录的本地副本。现在，社交网络的一个特征是用户可以通过自由的文本定制来更新自己的状态。比如，Alice可能想修改她的状态为“忙碌，电话中”(Bsy on the phone)，过后又修改为“有空，要聊天吗？”(Of the phone,anybody wanna chat?）。当Alice更改她的状态时，系统需要把这些更改操作写入数据库中她的用户信息记录，这样她的朋友们才可以看到。用户信息记录表见表4-1。注意，为了支持Web应用的不断升级，必须提供一个灵活的数据库模式和松散耦合的数据组织方式，而不是每条记录的每个字段都有相应值，并且增加新的字段代价必须小。

表4-1：用户信息表

Username	FullName	Location	Status	IM	BlogID	Photo	…
Alice	Alice Smith	Sunnyvale, CA	Off the phone, anybody wanna chat?	Alice345			…
Bob	Bob Jones	Singapore	Eating dinner		3411	*me.jpg*	…
Charles	Charles Adams	New York, New York	Sleeping		5539		…
…							

我们该如何更新Alice的用户信息记录？从标准数据库角度看，为了实现该更新操作的原子性，我们需要执行以下步骤：打开一个事务；对所有的副本执行写操作；向所有的副本发送提交信息并关闭该事务。这种方法和标准数据库的ACID[1]模式一致，保证所有的副本都能够正确地更新到新状态。即使是非ACID模式的数据库，如G公司的BigTable(Chang等2006），采用了类似的方法来同步更新数据的所有副本。但是，当存在地理空间上的数据复制时，该方法效率很低。当Alice输入她的状态信息并点击“OK”，由于需要等待分散在各地的数据中心提交该事务，她可能需要等待很长时间才能加载响应页面。此外，为了保证原子性，在事务处理过程中，系统需要对Alice的状态加排他锁，这意味着其他用户可能长时间无法看到Alice的状态。

由于当地理空间上存在不同副本时，原子事务的代价很高，很多Web数据库采用了“尽最大努力”的方式(bst-effort approach)：先将更新写到一份数据中，然后异步地将更新操作传播(popagate)到其他的副本。模拟事务操作时，不执行加锁及验证。这种方法，犹如其名——“尽最大努力”，存在很多困难。即使系统能够保证在所有的副本上执行更

新，也不能保证数据库的不同副本最终都保持一致性。比如，Alice首先将她的状态更新为“Busy”，这样在美国西海岸(wst coast)的数据中心需要执行写操作，如表4-2所示。

表4-2：在美国西海岸的数据副本上执行的更新操作

美国西海岸	
用户名	状态
Alice	Busy

美国东海岸	
用户名	状态
Alice	--

然后，Alice把她的状态更新为“Off the phone”，但由于网络干扰，该更新操作误在美国东海岸(est coast)的数据副本上执行，如表4-3所示。

表4-3：在东海岸数据副本上执行的第二次更新操作

美国西海岸	
用户名	状态
Alice	Busy

美国东海岸	
用户名	状态
Alice	Off the phone

由于更新传播是异步的，一个可能的事件序列如下：在美国东海岸的数据副本上首先执行“Off the phone”更新操作，然后执行“Busy”更新操作。于是，通过有线传播，更新操作如表4-4所示。

表4-4：传播中的两个更新交叉

美国西海岸	
用户名	状态
Alice	Busy

“Busy” →
← “Off the phone”

美国东海岸	
用户名	状态
Alice	Off the phone

在美国东海岸，“Busy”的状态覆盖了“Off the phone”状态；而在美国西海岸，“Off the phone”状态覆盖了“Busy”状态，这导致如表4-5所示的状态不一致性。

表4-5：不一致的数据副本

美国西海岸	
用户名	状态
Alice	Off the phone

美国东海岸	
用户名	状态
Alice	Busy

根据读取数据副本的不同，Alice的好友看到的Alice的状态也是不同的，而且该不一致性会一直存在，直到Alice再次改变她的状态。

为了解决上述问题，一些Web规模(wb-scale)的数据存储实现了最终一致性策(eentual consistency)，如上述的不正常现象可能暂时发生，数据库最终将会解决该不一致性，并保证所有的数据副本都有相同值。该策略是某些系统的核心，如Amazon的Web服务S3。最终一致性策略通常使用如闲话(gssip)和反熵(ati-entropy)的技术来实现。但是，即使数据库最终会收敛到某个值，也很难预测它最终的收敛结果。因为没有全局时钟来序列化所有的更新操作，数据库难以识别Alice的最终状态更新是“Busy”还是“Off the phone”，因此最后有可能会收敛为“Busy”状态。这样，正当Alice准备和好友聊天，她的所有好友都认为她是忙碌的，且在Alice再次更新她的状态前，这种不正常现象会一直存在。

我们的方法

强一致性策略（如ACID事务）存在扩展上的局限性，弱一致性策略（如尽最大努力策略或最终一致性策略）可能存在不正常现象，我们提出介于二者之间的一种策略，即时间轴一致(tmeline consistency)策略：所有的数据副本将通过相同的时间轴执行更新操作，而且该更新和数据库上执行的更新顺序相同。该时间轴如图4-1所示。因此，数据库将在

所有的数据副本上收敛为相同的结果值，且该值是该应用在数据库上执行的最近一次更新操作之后的值。

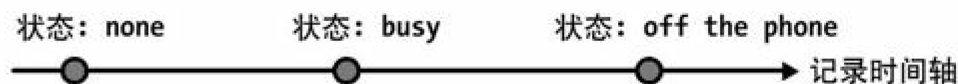


图　4-1：Alice状态的更新操作时间轴

时间轴一致性是这样实现的：由一台主备份服务器(mster copy，以下简称主备份）执行所有的更新操作，然后将变化异步传播到其他备份服务器来实现的。该主备份序列化所有的更新操作，并确保为每个更新指定一个序列号。即使在更新的异步传播中存在瞬间的故障或误序，该序列号的次序是和需要在所有副本上执行的更新次序一致的。我们选择每条记录一个主备份是因为雅虎的很多应用依赖于同一张数据表，该表中不同的记录对应不同的用户，每个用户都有不同的使用模式(uage patterns)。当然，有可能为主备份性(mstership)选择其他的粒度，比如每个记录分区（如基于关键字）有一个主备份。

即使是在同一张表，不同记录的主备份也可能分布在不同的服务器上。在以上例子中，Alice住在美国西海岸，并在美国西海岸的数据副本有一条主备份记录；而她的好友Bob住在新加坡，其主备份记录是在亚洲地区的副本。记录的主备份性是作为记录的元数据域(mtadata field)存储在该记录本身，如表4-6所示。

表4-6：包含主备份性和版本元数据的信息表

Username	_MASTER	_VERSION	FullName	…
Alice	West	32	Alice Smith	…
Bob	Asia	18	Bob Jones	…
Charles	East	15	Charles Adams	…
…				

主备份似乎和我们的原则——只应该同步执行代价低的操作不符。如果Alice去纽约旅行，并在那里更新她的状态，由于她的个人信息记录的主备份在美国西海岸，她必须等待系统把她的更新操作提交到美国西海岸；这种高延迟的跨州更新操作正是我们应该尽量避免的。由于用户转变使用模式（如Alice的旅行），这种跨数据中心的写操作确实会偶尔发生，但几率很小。我们分析了对雅虎的用户数据库的更新操作，发现记录更新在包含主备份的数据中心上执行的概率占85%。当然，Alice可能迁移到美国东海岸或者欧洲，由于其记录的主备份仍然在美国西海岸，她执行的写操作就不再是在本地执行的。我们的系统会追踪一条记录更新的源起，为了保证绝大部分的写操作还是在本地执行，系统会根据长期性访问模式(acess pattern)的转变来修改主备份性。（我们将在下一章更详细地讨论主备份性。）

当应用读取一条记录，它通常会读取该记录的本地副本。除非该副本标记为主备份，否则其数据可能已经过期了。该应用可以从时间轴识别其记录实例是和某个版本的记录一致，但是它无法从记录本身识别该记录是否为最新版本。如果该应用要求记录必须是最新版本，系统允许它发送最新读(u-to-date read)请求操作，该请求会被提交到主备份来获

取记录的最新备份。执行最新读操作代价很高，而通常情况下读本地副本（可能数据已经过期）的代价很低，这和系统的设计原则也是一致的。幸运的是，Web应用通常对数据时效性要求较低，能够接受过期数据。如果Alice更新了她的状态，但她的好友Bob没有马上看到，而是过后不久看到其新状态，也是可以接受的。

应用可以执行的另一类读操作是“临界读”(citical read)，可以保证从用户的角度来讲，数据只随时间向前移动。假设有这样的情况，Alice改变了她的头像（表示一个用户的图片），Bob可能会看Alice的信息页面（导致对数据库的一次读操作），并看到了Alice的新头像。然后，Bob可能会刷新该页面，但由于网络问题，该刷新操作被重定向到没有执行Alice头像更新的副本上执行。结果是Bob将看到的数据比他刚看到的版本更老。为了避免应用程序执行这些操作引起的这种不正常现象，数据库为读操作调用返回记录时会附上该记录的一个版本号。该版本号可以保存在Bob的会话状态(sssion state)或者其浏览器cookie中。当Bob刷新Alice的信息页面时，他上次读到的版本号将和本次请求一同发送，数据库会保证返回记录的版本不会老于该版本。这可能需要把版本号和请求提交给主备份。指定版本号的读操作被称为“临界读”，任何版本号等于指定版本号或者更新的副本，其返回的结果数据都是可以接受的。对于先更新再读取数据库的用户，该技巧尤其有用。以Alice自己为例，当她更新完头像后，如果有任何页面还显示其旧头像，她一定会感到很困惑。因此，当她执行了某个动作需要更新数据库（如更新头像）时，应

用程序可以使用“临界读”机制来保证页面都不会显示她的旧数据。

我们也支持“测试并设置”(tst-and-set)操作，该操作使得写操作必须依赖于当前读操作的版本和之前看到的版本（其版本号作为参数传递给“测试并设置”请求）相同。从传统的数据库系统考虑，这是ACID事务的一个特例，它限于单记录，使用的是乐观并发控制策略(otimistic concurrency control)。

主备份性的其他方面

我们采用了各种技术来保证即使在工作负荷变化、失效情况下，都能够低延迟地、平滑地执行读写操作。

举个例子，正如我们之前所述，该系统实现的是记录级(rcord-level)的主备份性。如果一条记录的很多写操作来源于某个数据中心而不是当前的主备份，该记录的主备份性将被及时地转移到该数据中心，且随后的写操作将在该数据中心本地执行。此外，转移主备份性代价很低且系统可以自动执行，因而系统可以快速地适应工作负荷的变化。

系统还实现了即使在存储单元失效的情况下，还能够不间断地执行读写操作的机制。当某个存储单元失效，会（手工或自动）执行对该存储单元的覆盖操作(oerride)，并指示另一个数据中心取代失效的存储单元，（为以该存储单元作为主备份的记录）执行写操作。系统会采取措施（细节不在这里赘述），确保根据在失效的存储单元已执行的更新操作执行顺序覆盖操作。这样，当其他的数据中心开始取代失效的存储单元执行更新操作时，还能够保证时间轴一致性。

在PNUTS中，所有读写请求会通过路由层来定位到一条记录相应的数据备份（可能是主备份）。该间接层(lvel of indirection)是提供不间断的系统可用性(uinterrupted system availability)的关键。即使当一个存储单元失效而其数据恢复到另一个存储单元，或者根据使用模式的变化转移记录的主备份，这些变化对于上层应用也是透明的，应用依然可以连接到路由器，享受不间断的系统可用性，其请求可以无缝地路由到适当的数据中心。

支持有序数据

PNUTS系统架构上同时支持散列分区和范围分区的数据。我们称散列版的数据库为YDHT，即雅虎分布式散列表(Yhoo!Distributed Hash Table)；称有序表版的数据库为YDOT，即雅虎分布式有序表(Yhoo!Distributed Ordered Table)。绝大多数系统不关心数据的组织方式。但是，有个重要的方面对物理数据的组织方式是敏感的。尤其是采用散列表方式组织的数据，往往可以把负载很均衡地分散到不同的服务器。如果数据是有序的，部分更频繁访问的关键字空间(ky space)将会出现“热点”(htpot)效应。比如，如果状态更新是通过时间排序，那么最近的更新是用户最感兴趣的，则包含在该段时间范围末的数据分区的服务器负荷会最重。在不影响系统向外扩展下，我们不允许“热点”效应持续。

逻辑有序的数据实际上是存储在物理相连的记录分区上，但是当分区无序组织时，这些数据就可能存储在多台物理服务器上。我们可以根

据负载，动态移动分区来解决“热点”效应问题。如果一台服务器上有多个热分区(ht partition)，可以把这些分区移到负荷较低的服务器上。此外，我们还可以动态划分分区，这样某个热分区上的负载可以分散到几台服务器上[2]。这种在存储单元间移动和划分分区的策略和先前提到的改变一条记录的主备份性机制是不同的：对于前者，改变记录的主备份会导致在一台服务器发起的更新延迟，但通常不影响给定记录分区记录的累积读写操作负荷。需要划分和移动分区的一个特例是当我们想要更新或插入大量的记录。在这种情况下，如果不采取特殊措施，就会发生只向几台服务器发送大批量更新操作，从而导致服务器负载不均衡。因此，需要理解更新操作是如何分布在关键字空间(ky space)，必要的时候，可以为后期的一连串更新操作预先划分和移动分区(Slberstein等2008）。

我们把应用和物理数据组织细节相分离。对于单记录的读写操作，使用路由层可以使应用免于受分区移动和划分的影响。对于范围扫描，我们需要提供进一步抽象：假设我们需要扫描年龄在21～30岁的所有注册用户，执行该查询可能意味着在一台服务器上需要扫描包含几千条记录的数据分区，在另一台服务器上扫描另一个分区等。每个分区有几千条记录，由于它们在磁盘上是顺序存储的，因此可以快速扫描。我们希望这些分区的移动和划分对于应用是透明的。解决该问题的一个巧妙方式是借鉴迭代器(ierator)的思想：当应用执行扫描时，我们返回一组记录，然后准备好下一组记录，触发应用继续扫描。因此，当应用执行完

一次批处理，开始请求更多的数据时，可以把这些请求转换到包含下一组记录分区的新存储服务器上执行。

牺牲一致性，换取可用性

时间轴一致性处理一般问题效率高，且语义简洁，但它也并不完美。有时，整个数据中心崩溃（如断电）或者不可用（如断网），那么以该数据中心作为主备份的记录将不可写。这种情况暴露了已有的在一致性、可用性和分区容忍性(prtition tolerance)上的平衡问题(tade-off)：在所有情况下，这三者属性只能保证其中的两个。由于数据库是全球性的，分区必然存在且不会造成大的影响，因此实际上我们只需要处理一致性和可用性间的平衡。如果一个数据中心断网了，可能有些新的更新操作还没有传播到其他的数据副本，我们可以通过两种方式保证一致性：一是为了保持一致性，在该数据中心可用前，不允许执行数据更新；二是为了保持可用性，要违背时间轴一致性原则，允许一些更新操作应用到非主备份的记录。

系统为应用提供了基于表级粒度(pr-table basis)来选择相应策略。如果对于某个特定的表，可用性对于该应用的优先级比一致性高，那么当一个数据中心断网时，系统会临时转换该表中任何不可用记录的主备份性到其他的数据中心。该决策牺牲时间轴一致性，有效地支持了可用性，如图4-2中的例子所示。当恢复完一个不可用的数据中心，系统会自动协调任何有冲突的更新记录，并通知应用这些冲突信息。该协调保证了即使不能满足时间轴一致性，数据库在任何数据备份上都能收敛到

相同的值。另一方面，对于应用而言，如果一致性优先级比可用性高，则会保持时间轴一致性而不会传递主备份性，这样，一些写操作执行会失败。

对于某些操作，一致性和可用性间的平衡较易于把握。例如，假设有个关于投票的应用，用户可以对各种问题进行投票（如“你最喜欢的颜色是什么？”），投票结果作为计数存储在数据库上。计数操作（如增加）是可交换的(cmmutative)，因此，在不破坏时间轴一致性的情况下，甚至可以应用于非主备份副本上。一般来说，复制机制可以在不同副本间传递记录的新版本，但是对于可交换操作，我们实际上是传递操作（如增加）而不是结果值。那么，主备份在任何时间接收到操作（不论是在正常的数据操作期间还是在某个数据中心失效后），它都可以执行该操作，而不需要担心操作是否乱序。该策略的一个局限是可交换和非可交换的操作不可以混合：因为我们无法知道如何合理地对一个计数值的增加操作和重写操作进行排序，所以当记录插入后，禁止在任何时刻对其计数器设值。

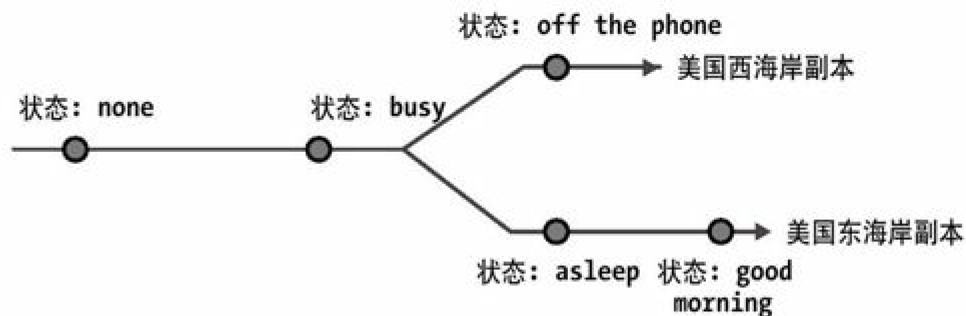


图　4-2：西部数据中心断网，更新时间轴产生交叉(frks)

PNUTS的另一个扩展是允许对多条记录执行更新操作。我们主要研究基于记录级的时间轴一致性的原因是很多Web应用负载涉及在某一时刻对单条记录的更新操作。但是，有时更新多条记录更合适。例如，在社会网络应用中，可能存在双向好友链接：如果Alice和Bob是好友，那么Alice会出现在Bob的好友列表，同样，Bob也会出现在Alice的好友列表。因此，当Alice和Bob成为好友时，我们需要更新两条记录。因为系统不提供ACID事务，所以无法保证该更新是原子性的。但是，我们可以采取绑定式写操作(bndled writes)：当对数据库执行一次调用时，应用可以请求两个写操作，数据库会保证两个写操作最终都会执行。为了实现这点，系统对请求的写操作写日志，如果初始执行失败，系统会重试这两个写操作，直到它们都成功。该方法保证了记录级的时间轴一致性，由于重试可以异步执行，这也保证了系统的性能。

总之，时间轴一致性为记录更新如何传播提供了简单的语义支持(smantics)，以及为应用如何牺牲读延迟来换取流通性(crrency)提供了灵活性。但是，它并不支持一般的ACID事务，尤其是执行多条记录的读写操作事务。

[1]事务具有原子性、一致性、隔离性（不受其他并发事务影响）和持久性。

[2]细心的读者可能会发现，如果所有的更新影响包含该时间范围内的分区，划分该分区并不能解决问题，需要一些其他方法，如通过组合关键字如用户名和时间进行排序。

复杂查询

随着Web应用更加复杂有趣，这些应用需要以新的不同方式从数据库中检索和组合信息。下一步，我们将探索如何在大规模系统上支持这些查询。

面临挑战

系统对只需要一条或者几条记录的查询做了优化。尤其是，系统可以通过主键来查询记录；只要知道Alice的用户名，就可以直接确定哪个分区包含她的信息记录，加载页面时读取该记录。另外，系统可以采用散列分区表或范围分区表来存储数据。对于范围分区表，可以通过主键的排序范围做范围扫描。例如，系统可能以每个连接一条记录的方式存储Alice的好友列表，其中每个连接的主键是Alice和她好友的用户ID对（见表4-7）。

表4-7：好友关系表

User1	User2	…
Alice	Bob	…
Alice	Charles	…
Alice	Dave	…
…		

对于范围分区表，所有以“Alice”为前缀的记录会聚集在一起，因此，只需要小范围扫描就可以检索到这些相关记录。

现在假设我们需要给社交网络站点添加另一个功能。在该站点中，用户可以发布照片，对其他人的照片发表评论。Alice可能想对Bob、Charles和Dave的照片发表评论。当页面显示一张照片，我们希望它也能够显示和该照片相关的一组用户评论，同时还想为Alice显示她对其他人的照片做出的一组评论。评论表的主键设定为(PotoID,CommentID)，并以有序的YDOT表的方式存储（见表4-8），因此对相同照片的所有评论会聚集在一起，可以通过范围查询来检索。

表4-8：照片评论表

PhotoID	CommentID	Comment	Commenter
Photo123	18	Cool	Mary
Photo123	22	Pretty	Alice
Photo123	29	Interesting	Charles
…			

系统如何收集Alice做出的所有评论？它需要对Alice的信息记录（以username为主键）和评论记录（以username做外键）做连接查询。由于向外扩展的系统架构，数据分区跨越很多服务器，因此连接操作需要访问很多服务器。该操作一方面需要连接多台服务器，另一方面一条查询占用大量的服务器资源（导致其他的请求响应变慢），因此代价很高，增加了请求延迟。

在向外扩展的系统中，另一种代价高的操作是聚集(goup-by-aggregate)查询。假定用户指定了爱好，系统想要计算每种爱好的用户数，从而可以向Alice展示哪种爱好是最受欢迎的。这种查询需要扫描所有的数据并保存计数值。这种表扫描操作会给系统带来巨大的负荷，而且不能够同步执行，因此Alice的页面需要非常长的时间才能加载。

这些例子说明了虽然单点查询和范围扫描可以快速执行，但是代价更高的连接和聚集查询操作不能够同步执行。

我们的方法

处理代价高的操作，核心原则是异步执行，但是代价高的查询实际上不能够以这种方式处理；我们不希望需要Alice不断返回检查获取其所有评论的异步查询是否已经执行完成。

实例化视图可以通过异步式构建(Aarawal等2009），当Alice登录时，她可以快速（同步）地查询该视图[1]。虽然和基本表的数据相比，异步式构建的视图可能已经过期，但由于应用本来就需要处理过期的副本，所以处理过期的视图数据通常是可以接受的。事实上，我们把实例化视图作为一种特殊的副本，该副本可以复制数据和转换数据。通过与更新副本相同的机制来更新视图，系统保证了视图和复制的基本表数据，有相似的可靠性和一致性保证，而不需要再设计和实现另一个机制。

虽然视图是在后台维护，但是我们还是希望尽量减少它的代价。如果维护视图占用很多系统资源，或者它会影响同步的读写请求（增加每

个查询的延迟），或者会被分配很少的资源，运行很慢，这样视图数据会严重过期而导致视图可能毫无用处。因此，我们必须探索使视图维护高效的策略。比如对于先例，系统想给Alice显示她对其他人的照片做出的所有评论。系统将创建一个实例化视图，该视图通过外键（评论员的username)而不是主键对评论数据进行重组聚合。那么，Alice做出的所有评论会聚合在一起，我们还可以给视图添加Alice的信息记录，以她的username为主键，因此Alice的信息和评论可以聚合在一起。主键/外键连接操作如同扫描并连接以“Alice”为前缀的视图记录一样简单。结果如表4-9所示。

表4-9：信息和评论记录的连接和聚合

Alice	West	32	Alice Smith	…	← 信息记录
Alice	Photo123	22	Pretty	…	← 评论记录
Alice	Photo203	43	Nice	…	↓
Alice	Photo418	33	OK	…	
…					

注意，我们并没有在视图中做信息和评论记录的预连接操作，只是通过定位需要连接的记录，使得连接的维护代价低。当需要对基本表记录做更新操作时，我们只需要更新一条视图记录，即使该视图将和很多其他记录做连接操作。

我们如何在同一个表存储信息和评论记录？在传统的数据库，因为两种记录有不同的模式(shema)，所以这样会很难存储。但是，PNUTS的一个核心特征是能够表示灵活的数据库模式。相同表中不同的记录可

以有不同的属性组。这个特性对于Web应用特别有用，因为Web数据通常稀疏松散，一个销售商品的数据库根据商品的种类，会有不同的属性（如color、weight、RAM和flavor)。灵活的数据模式对于实现实例化视图连接也是非常重要的，这样我们可以定位来自不同表的连接记录。

异步式视图方法也有助于响应其他类型的查询。聚集查询可以通过实例化视图预先分组(pe-group)，甚至可以预先聚合(pe-ag gregate)数据，因此可以有效地响应查询。甚至有诸如选择非主键属性的“简单”查询，这种查询通过实例化视图方式处理最有效。假设需要查询住在加州Sunnyvale的所有用户。因为用户表的主键是username，该查询通常需要执行代价高的表扫描操作。但是，我们可以采用实例化视图在用户表的“location”域构建索引，把该索引存储到有序的YDOT表，然后在“Sunnyvale,California”的索引记录上执行范围扫描来获取结果，从而响应查询请求（见表4-10）。

表4-10：地理位置索引

Location	Username
Sunnyvale, CA	Alice
Sunnyvale, CA	Mary
Sunnyvale, CA	Steve
Sunnyvale, CA	Zach
…	

如其他系统中的实例化视图，只要我们事先知道需要什么查询，就

可以有效地创建视图。对于提供Web服务的应用，查询通常是根据事先确定的模板，在运行时指定参数（如地理位置或用户名）。因此，应用开发者可以预先知道哪些查询很复杂，需要实例化视图。为了即时请求存储在PNUTS中的数据，开发者需要使用PNUTS插件从系统中获取数据，并把数据保存到运行着Hadoop的计算网格。Hadoop是MapReduce的开源实现。当有一些不同的机制来处理复杂的查询，实现有助于有效执行查询的查询计划器是非常有用的。查询计划器可以减轻应用开发者的负担，它可以写声明性查询(dclarative queries)，而不需要考虑这些查询将如何执行。但是，对于如PNUTS的大规模系统，有效的查询计划器需要复杂的统计收集、负载监测、网络监测和一系列其他机制来保证查询计划器对于系统的所有可能瓶颈有足够的信息来制定最有效的查询计划。

[1]在PNUTS系统的生产版本中，还不包含实例化视图。

和其他系统的比较

当我们开始思考PNUTS时，G公司和Amazon已经宣布了两个大规模可扩展性数据库系统，微软也将对外宣布。当我们开始设计时，我们仔细研究了这些系统，思考它们的思想是否对我们有用。这些系统的一些想法对我们有影响，但我们还是决定构建一个新的系统，其架构在很多方面和这些系统不同。我们现在先来研究这些系统，并讨论为什么决定不采用它们的设计理念。

G公司的BigTable

BigTable(Chang等2006）是为了支持G公司的很多Web应用而设计的系统。该系统基于水平方式将“大表”(bg table)切分为很多小表(tblet)，并把这些小表分散到不同的服务器上。支持可扩展性的基本方法以及一些其他特性，比如灵活的模式和有序存储，与我们采用的方法很相似。但是，我们的设计和BigTable在某些方面有区别。

首要区别是数据复制方式不同。BigTable是构建在G公司的文件系统GFS上(Gemawat等2003）。GFS通过同步更新三个数据备份到三台不同的服务器上的方式来处理数据备份。这种方法在一个数据中心可以工作良好，因为一个数据中心的服务器间的延迟很短。但是，在三个不同的、分散的数据中心同步更新三台服务器代价很高。Alice可能需要等待很长的时间，然后其状态才能更新，尤其是如果她的好友通过带宽很低的网络来访问数据中心。为了支持跨数据中心的数据复制，我们提出了

时间轴一致性模式，以及与主备份性、负载平衡和失效处理相关的机制。

BigTable和GFS在数据库服务器和文件系统间存在分离，而PNUTS不采用该分离。GFS最初是为需要对大文件进行扫描的应用设计和优化的（如MapReduce)。BigTable通过保留每个记录的版本历史和GFS关联，并把文件以SSTables的文件格式压缩来减少存储空间。这意味着对于记录的读和更新操作，数据需要采用这种压缩格式解码和编码。此外，GFS的特征是面向扫描，这有助于BigTable处理面向列的扫描应用（如“返回所有用户的所有地理位置信息”）。与之相反，我们的主要应用是需要读和更新一个小范围记录内的一条记录的一个版本。因此，我们的策略是把数据作为完整的记录，以B树形式组织保存在磁盘上。这种方法对快速定位、就地更新由主键识别的各条记录做了优化。

PNUTS和BigTable在其他方面还有不同。比如，PNUTS对于一个应用支持多个表，而不是一个大表，且对于散列表和有序表都支持。MegaStore是继BigTable之后(Frman等，2008）添加了事务、索引和更丰富的API，但仍然遵循BigTable中的基本架构原则。

Amazon的Dynamo

Dynamo(DeCandia等2007）是Amazon最近构建的应用于大规模数据负荷的系统，是和我们的高可用性、大规模数据存储的目标最相近的一个系统(Dnamo中称记录为对象）。Dynamo通过允许应用向任何副本写数据提供了写操作可用性，它通过Gossip协议（稍后会细述）向其他副

本延缓传播(lzily propagate)这些更新操作。

为了解决网速慢和易于失效问题，Dynamo和PNUTS一致，它们都采用了延缓传播更新策略(lzily propagate updates)。但是，PNUTS的复制策略与之有很大区别。在Gossip（闲话）协议中，更新操作会传播到随机选定的副本，该副本又把该操作继续传播到其他随机选定的副本。这种随机性是该协议提供的概率性保证的基础，它保证了绝大多数副本可以相对快速更新。然而，在我们的应用环境中，随机性必然是次优的。比如Alice执行状态更新操作，其数据是在美国西海岸的数据中心。若采用Gossip方式，该操作可能随机传播到新加坡的副本，然后又随机传播到Texas的副本，接着又随机传播到东京的副本。该更新操作在传播地域上跨越了三次太平洋，如果采用一种更确定性方式(dterministic approach)，只需要传播该操作（和其他更新操作）一次，可以节约紧缺的跨太平洋主干带宽。此外，Gossip要求传播数据的副本预先知道在其他哪个数据中心的哪台服务器上有副本，因此很难为了负载均衡或数据恢复，在不同服务器间传送数据。

PNUTS和Dynamo的另一个关键区别是一致性协议。Gossip协议的本质决定了它是最终一致性模型：所有的数据副本最终将保持一致，但是在更新传播期间，副本可能是不一致的。尤其是，副本可能会存在“无效”状态。举个例子，Alice把她的状态从“Sleeping”改为“Busy”，再把地理位置信息从“Home”改为“Work”。由于更新的次序，从Alice的角度出发，唯一有效的记录状态是(Seeping,Home)、(Bsy,Home)和

(Bsy,Work)。在最终一致性原则下，如果两个更新是在不同的数据副本上执行的，有些副本可能会首先接收到改为“Work”的更新，这样，这些副本将临时显示(Seeping,Work)状态。如果Alice的老板看到这种状态，Alice可能就有麻烦了。对依赖于在一条记录上以合理的次序执行多次更新的应用，它需要比最终一致性更严格的执行次序保证。虽然时间轴一致性模式允许副本已经过期，但即使是过期的副本也应该是和某个合理的更新次序一致的版本。

PNUTS和Dynamo还有很多其他区别：Dynamo只支持散列表，不支持有序表，PNUTS二者都支持；和Dynamo相比，PNUTS支持更灵活的数据到服务器的映射机制，可以提高负载均衡和数据恢复（尤其是对于有序表，可能存在不可预测的“热点”）。除了DynamO'Amazon还提供其他的存储系统：S3用于存储数据块；SimpleDB用于在结构化、包含索引的数据上执行复杂查询。虽然SimpleDB提供了更丰富的API，但是它需要应用实现某种数据分区策略，使得每个分区都有固定的大小限制。因此，一个分区内的数据增长是受限的。

微软的Azure SDS

作为Azure服务提供的一部分，微软构建了大规模的SQL Server服务器，称为SQL数据服务(SL Data Service或SDS，参见
http：//hadoop.apache.org)。SDS的核心是通过水平分区达到系统的可扩展性。SDS的一个良好的特性是它通过扩展数据索引，并以SQL Server作为查询处理引擎，提高了查询能力。然而，SDS通过严格的分区来实

现其查询表达能力：如果应用创建自己的分区，则无法简单地进行重分区。因此，虽然可以对一个数据分区发出丰富的查询请求，如果某个数据分区数据量增长或者访问很多，那么系统无法简单或自动地通过划分分区来解决“热点”效应。我们的策略是通过表抽象层，使得为加载和恢复数据而做的数据分区或者改变这些分区对于上层应用是透明的。虽然PNUTS的查询模式表达能力不如SDS(PNUTS不支持跨分区的复杂查询），但是我们还在继续探索增强PNUTS查询功能（如前面描述的通过视图的方式）的策略。

PNUTS和SDS的另一个区别是PNUTS系统把地理复制作为内建的主要特征。至少对于SDS的第一版本，其工作负荷是期望全部在一个数据中心，只有当主副本全部崩溃时，才会使用远程备份。我们希望Alice在新加坡、柏林和里约热内卢（巴西）的好友都能够有他们自己本地的Alice更新的首要备份。

其他相关系统

和我们的系统类似，一些需要扩展性和灵活性的其他公司，也构建了很多其他系统。Facebook构建了Cassandra系统(Lkshman等2008），它是一个P2P(Peer-to-Peer，对等网络）的数据存储，其数据模型类似于BigTable，架构类似于Dynamo，系统只提供最终一致性。

Sharded数据库（如Flickr[Pattishall]和Facebook[Sobel 2008]使用的MySQL的分区(sarding)策略）通过把数据分区到很多服务器上来提供可扩展性。但是，Sharding系统并没有提供我们所期望的扩展灵活性或全

球数据复制特征。和SimpleDB类似，Sharded数据库的数据需要预分区。此外，该数据库只有一个副本可以是主备份，执行写操作。在PNUTS中，不同数据中心的所有副本都可以执行写操作（虽然是对不同的记录）。

雅虎的其他系统

PNUTS是雅虎构建的众多云系统中的一个。在数据管理中，还包含云计算的其他两个方面，但是研究的问题和PNUTS不同。Hadoop是MapReduce框架(Dan和Ghemawat2007）的开源实现(htp：//hadoop.apache.org)，它提供了在大数据文件上的大规模并行分析处理。Hadoop的文件系统HDFS，对扫描操作做了优化，因为MapReduce主要是处理面向扫描的操作。与之相反，PNUTS主要是处理单条记录读写操作。另一个系统MObStor是为了存储和服务大数据对象如图片和视频而设计的。MObStor的目标是为静态对象提供低延迟检索和低代价存储。因为很多应用需要结合记录存储、数据分析和对象检索，我们正在探索无缝地集成这三个系统(Coper等2009）。

结论

当启动PNUTS项目时，我们希望该系统能够无缝地扩展到跨域多个州的成千上万的服务器。构建这样的系统不仅需要巧妙的工程技术，也要求我们在数据库很多已有的领域继续探索。虽然放弃ACID特性是相对轻松的决策，但是我们很快发现需要提出其他的方案来取代它，因此我们提出了时间轴一致性模型。虽然该模型在设计上相对简单，但处理复杂的极端案例(crner cases)，提出有效的实现机制以及构建应用用例模型都需要反复深入的思考。值得一提的是，刚开始我们的客户和我们自己都不太关心该系统只支持简单的查询语言的不足。但是，随着开发者开始试着在PNUTS上构建真正的应用，我们意识到该系统不能处理的小部分复杂查询将影响该系统被采用。如果我们不提出解决这些问题的机制，那么开发者就需要采用一些复杂的变通方案，在应用内部实现代价高的操作（如嵌套连接）或者是频繁地将数据导出到其他的索引来支持他们应用的工作负荷。

该领域是云数据管理的新兴领域，这在很多其他可选系统设计的构建和部署中也体现出来。我们希望PNUTS的思想有助于我们更接近构建易于管理、广泛应用、多租户(mltitenanted)云数据库系统的目标，为应用提供弹性、高效、全球可用的和无比健壮的数据后台。

致谢

PNUTS是雅虎很多员工共同努力的结果。工程方面的负责人是P.P.S.Narayan和Chuck Neerdaels。该项目的其他研究员包括Adam Silberstein和Rodrigo Fonseca。Brad McMillen和Pat Quaid帮助构建PNUTS架构并把它用于雅虎的云计算服务提供中。该系统的其他设计和开发人员包括Phil Bohannon、Ramana Yerneni、Daniel Weaver、Michael Bigby、Nicholas Puz、Hans-Arno Jacobsen、Bryan Call和Andrew Feng。

参考文献

[1]Azure Services Platform. http：//www.microsoft.com/azure/.

[2]Hadoop. http：//hadoop.apache.org.

[3]Agrawal,P.， A. Silberstein,B.F.Cooper,U.Srivastava,and R.Ramakrishnan."Asynchronous View Maintenance for VLSD Databases."In SIGMOD，2009.

[4]Chang,F. et al."Bigtable:A distributed storage system for structured data."In OSDI，2006.

[5]Cooper,B. F.， E.Baldeschwieler,R.Fonseca,J.J.Kistler,P.P.S.Narayan,Chuck Neerdaels,Toby Negrin,Raghu Ramakrishnan,Adam Silberstein,Utkarsh Srivastava,and Raymie Stata."Building a cloud for Yahoo！"IEEE Data Engineering Bulletin，32（1）：36-43，2009.

[6]Cooper,B. F.， R.Ramakrishnan,U.Srivastava,A.Silberstein,P.Bohannon,H.-A.Jacobsen,N.Puz,D.Weaver,and R.Yerneni."PNUTS:Yahoo！'s hosted data serving platform."In VLDB，2008.

[7]Dean,J. and S.Ghemawat."MapReduce:Simplified data processing on large clusters."In OSDI，2004.

[8]DeCandia,G. et al."Dynamo:Amazon's highly available key-value

store."In SOSP，2007.

[9]Furman,J. J.，J.S.Karlsson,J.-M.Leon,A.Lloyd,S.Newman and P.Zeyliger."Megastore:A Scalable Data System for User Facing Applications."In SIGMOD，2008.

[10]Ghemawat,S.，H. Gobioff,and S.-T.Leung."The Google File System."In SOSP，2003.[11]Lakshman,A.，P.Malik,and K.Ranganathan."Cassandra:A Structured Storage System on a P2P Network."In SIGMOD，2008.

[12]Pattishall,D. V."Federation at Flickr:Doing Billions of Queries Per Day."http：//www.scribd.com/doc/2592098/DVPmysqlucFederation-at-Flickr-Doing-Billions-of-Queries-Per-Day.

[13]Ramakrishnan,R. and J.Gehrke.Database Management Systems.McGraw-Hill,New York,NY，2002.

[14]Silberstein,A.，B. F.Cooper,U.Srivastava,E.Vee,R.Yerneni,and R.Ramakrishnan."Efficient bulk insertion into a distributed ordered table."In SIGMOD，2008.

[15]Sobel,J."Scaling out."Facebook Engineering Blog,August 2008.

第5章　信息平台和数据科学家的兴起

Jeff Hammerbacher

图书馆和大脑

在我17岁时，我丢掉了在印第安纳州Fort Wayne的Scott杂货铺的出纳员工作。在我上大学前仅仅两个月里，我看到了没有工作带来的机遇。我没有告诉父母自己被解雇了。每天下午，我依然穿着出纳员的工作服离开家：黑色裤子、黑色皮鞋、白色衬衫，还有罩衫。在父母看来，我这身穿着是为严谨的账单审查工作准备，实际上，我是要在公共图书馆看10个小时的书。

所有好奇心强的人都想知道大脑是如何工作的，17岁的我更是超乎寻常的好奇。我在图书馆里学习大脑如何工作、休息和重建。除了使我们保持平衡、调整体温、不时地眨眨眼，大脑还摄取、处理和生成大量的信息。我们对周围的环境产生无意识的条件反射，养成短期口头禅和肢体特征，做出择偶和教育的长期计划。大脑令人感兴趣的不仅仅是它对感官数据做出反应的能力，而是作为信息库，生成计划和创建新的信息。我很想知道它是如何工作的。

然而，大脑的特点是其存储的信息只在一个人身上。为了从很多大脑收集信息，我们建造了图书馆。为了今后的利用，图书馆科学领域已经为图书馆的信息存储发展了众多的技术。关于该课题的一个有趣的读本是Alex Wright的《Glut》(Jseph Henry出版）。除了为今后检索存储信息，图书馆在创建新信息方面也起了重大作用。正如哲学家Daniel Dennett所说的："学者即是以图书馆的方式创建另一个图书馆"（a

scholar is just a library's way of making another library)。

图书馆和大脑是信息平台的两个例子。它们是组织进行摄取、处理和生成信息的场所，它们加速了从经验数据中学习的过程。当我在2006年加入Facebook时，很自然地开始构建了一个信息平台。因为Facebook用户数量剧增，我们团队构建的系统最终需要管理几十亿兆字节(ptabyte,PB)[1]的数据。在本章中，我将详细阐述在构建Facebook信息平台遇到的挑战，以及在开源软件上构建解决方案过程中汲取的教训。我还会概述在利用信息构建数据密集型产品和服务，并且帮助整个企业制定、实现目标过程中，数据科学家所起到的重大作用。在整个过程中，我还会描述一些其他企业在过去几十年里如何构建信息平台来处理这些问题。

在开始介绍前，首先需要指出的是我去图书馆而不是去杂货店的计划还是很快泡汤了。自在地读了几天后，某天晚上我从图书馆出来时，却找不到车子了。对我来说，在那时丢车很平常，但是停车场是空的，所以我知道出事了；最后是母亲发现了我的伎俩，把我的车拖走了。在走回家的漫长路上，我在心里学会了一课：应该带着怀疑的态度来考虑自己的策略，另外，不要和母亲斗智。

[1]1PB(petabyte)=1024TB，1TB(terabyte)=1024GB

Facebook具有了“自知之明”

在2005年9月，Facebook首次向非大学生公开，允许高中生注册账号。忠实的用户愤怒了，但Facebook团队认为这是为网站做出的正常方向。那么它该如何证明它的方案是正确的呢？

此外，在几乎所有可登录Facebook网站的学校中，Facebook已经渗入学生当中，但还是在有部分学校中，该网站一直不受青睐。和那些更成功的网络相比，这些落后的网络对于Facebook有什么区别呢？Facebook团队应该如何做才能激励他们的成功？

当我在2006年2月参加Facebook面试时，他们正积极地期望找到这些问题的答案。我曾在大学学习数学，在华尔街工作近一年，工作内容是构建模型来预测利率、价格复杂的衍生产品和对冲抵押贷款池；有一定编程经验，GPA成绩“暗淡”。虽然我的背景可能不太理想，但是Facebook却给了我研究科学家的职位。

几乎同时，Facebook聘用了一个报告分析主管。该主管在解决问题方面的经验远远超过我。我们和另外一个工程师一起，开始着手构建一个数据收集和存储平台，以便找到我们产品以上问题的答案。

我们第一个尝试是构建一个离线信息库，其涉及两个方面：一是用Python脚本把查询分发到Facebook的MySQL服务器层，二是采用C++实现守护进程，实时地处理事件日志。当脚本可以如期运行，我们每天收集大约10GB的数据。我后来明白系统的这部分通常称为“ETL”过程，即

抽取、转换和加载。

Python脚本和C++守护进程从Facebook的数据源系统中抽取数据，然后这些数据又被加载到MySQL数据库用于离线查询。我们在包含这些数据的MySQL上又运行了一些脚本和查询，对数据进行聚集，以便得到更有用的表现方式。这种用于决策支持的离线数据库即“数据仓库”(Dta Warehouse)。

最后，通过简单的PHP脚本把数据从离线的MySQL数据库抽取出来，向内部用户展示收集到的信息摘要(smmary)。这是我们第一次可以回答网站特性对用户行为的影响。早期通过以下几种渠道分析最大化增长：登出用户的默认页面的布局、邀请来源、Email联系方式导入器的设计。除了以上分析，我们开始通过历史数据开发简单的产品，包括对赞助商成员特性进行聚集的内部项目。实践证明，该项目很受品牌广告商欢迎。

我那时没有意识到，实际上，通过ETL框架、数据仓库和内部控制台(dshboard)，我们已经构建了一个简单的“商业智能”(Bsiness Intelligence)系统。

商业智能系统

在1958年IBM系统期刊的一篇论文中，Hans Peter Luhn描述了一个系统，该系统基于每个用户“动作点”(ation points)的“兴趣信息”(iterest profiles)，将文件通过“选择性传播”(slective dissemination)到动作点上。作者描绘了令人震撼的预测科学理论。该文章的标题是“一个商业智能系统”，它是“商业智能”这一术语在现代环境中的第一次应用。

除了实时信息传播，该系统还可以进行“信息检索”，即搜索，可以在全文资料库中执行。Luhn强调动作点，突出了信息处理角色在完成任务中的重要性。换句话说，仅仅收集并聚集数据是不够的；由于利用数据进行推断的洞察力很重要，企业必须提高能力来完成艰巨的任务。他还提出“报告人”需要周期性地筛选数据，有选择地将信息移到需要的动作点上。

从Luhn的论文发表之后，商业智能领域在过去五年得到不断发展，“商业智能”这个术语和结构化数据的管理更加紧密相关。今天，经典的商业智能系统是由ETL框架和商业智能工具组成：ETL有规律地把一组数据源中的数据抽取出来并加载到数据仓库中；商业分析师利用商业智能工具在该数据仓库上生成报告供内部使用。那么，我们是如何从Luhn的愿景走到今天？

在1970年，E.F.Codd首先提出了关系模型；在20世纪70年代中期，IBM拥有了关系数据库管理系统(RBMS)的工作原型。RDBMS极大地促

进了构建面向用户的应用，到了20世纪80年代早期，这些应用已经非常普遍。

在1983年，Teradata出售了第一个关系数据库，该数据库是专门为Wells Fargo公司的决策支持而设计。几年后，在1986年，Ralph Kimball成立了Red Brick Systems公司，面向相同的市场构建数据库。以后，应用开发商就使用Teradata和Red Brick系统来开发解决方案，但是直到1991年，在数据仓库方面的首篇经典论文才发表。

Bill Inmon的《Building the Data Warehouse》(Wley出版社）一书，对数据仓库设计做了连贯清晰地阐述，它包含构建数据仓库的详细方法和最佳实践。Inmon提倡先仔细研究已有的数据源和商业目标，再构建企业数据模型。

在1995年，随着Inmon的书越来越受欢迎，数据仓库在企业数据中心内大量繁衍，数据仓库机构TDWI(The Data Warehouse Institute)诞生了。TDWI组织了有关会议和报告，在阐述和传播数据仓库方面的知识上，其依然是个核心力量。在斯坦福大学启动它的WHIPS研究机构那年，数据仓库在学术领域开始传播。

1996年，Ralph Kimball发布了《The Data Warehouse Toolkit》(Wley出版社）一书，对Inmon理论进行了挑战。Kimball提出了实现数据仓库涅(nrvana)的不同方式，首先摒弃了企业数据模型。Kimball认为不同的企业单位应该构建他们自己的数据“集市”(mrts)，这些集市可以通过“总线”(bs)相连。此外，Kimball提倡使用多维建模，而不是使用范化的数

据模型，前者的关系数据模型可以通过少量人工干预来适应很多数据仓库实现中的特定的工作负荷。

随着数据仓库的增长，通常情况下，商业分析家想要快速地操作少量的数据子集，通常这些数据是通过几个“维度”参数来描述。基于这些观察，微软的一组研究人员，包括Jim Gray，在1997年推出了CUBE操作符。该新的操作符允许在小的、多维的数据集上进行快速查询。

维度模型和CUBE操作符都意味着虽然关系模型在构建面向用户的应用方面获得了成功，但它可能并不是构建信息平台的最佳方案。此外，文档(dcument)和动作点(ation point)是Luhn关于商业智能系统提议的核心，而不是表。另一方面，整个工程师团队在构建关系数据处理系统中具备出色的专业知识。

在分析了一些背景知识后，我们再一起回到关于Facebook的探讨。

数据仓库的消亡和重起

在Facebook，我们不断地向MySQL数据仓库加载更多的数据，执行更多的查询。我们只执行过为动态网站提供服务的数据库查询，对于在数据仓库上执行一条查询所需运行的时间之长，令我们都很惊讶。在和一些经验丰富的数据仓库专家讨论后，我明白了由于查询复杂性、数据量大或者两者兼备，查询执行几个小时甚至几天都是很正常的。

有一天，我们的数据库数据接近100万兆（1TB)，mysqld守护进程突然中断了。在诊断了一段时间后，我们试着重启数据库，熬了一宿，直到重启操作开始执行，我们才回家。

当我次日早上回去工作时，数据库还处在恢复阶段。由于数据被很多客户端修改，为了获得一致性视图，数据库服务器维护了一张持久性列表，该列表包含了所有的修改操作，称为“重做日志”(rdo log)或者“预写日志”(wite-ahead log)。如果数据库服务器被粗暴地终止和重启，它将会从重做日志中读取最近的写操作日志，加快恢复速度。由于我们的数据仓库系统规模很大，MySQL数据库恢复需要一定的时间。在那次崩溃之后，我们花了三天时间才重新拥有可以正常工作的数据仓库。

我们那时决定把数据仓库迁移到Oracle上，Oracle数据库软件对于管理大数据集有更好的支持。我们还购买了一些昂贵的高密度存储服务器和一个强大的Sun服务器来运行新的数据仓库。

在我们把程序从MySQL迁移到Oracle的过程中，我充分体验到所谓

的标准关系数据库在实现上还是存在许多差别。每个数据库的大批量导入和导出工具使用完全不同的机制。此外，每个数据库支持的SQL语句很不一样，这迫使我们不得不重写很多查询。更糟的是，Oracle上的Python客户端库还不正式且有些bug，因此遇到问题时，我们需要直接和开发人员联系。

经过几周的辛苦努力，我们重写的脚本在新的Oracle平台上可以运行了。夜间执行的程序运行正常，我们很兴奋地去尝试Oracle“生态系统”的其他工具。尤其是，Oracle有个名为Oracle仓库构建器(Oacle Warehouse Builder,OWB)的ETL工具，我们期望用它替代自己手工写的Python脚本。但是，该软件不支持我们需要的数据源规模：在那时，通过每天晚上收集数据，Facebook已经有几万个MySQL数据库。虽然连Oracle都无法帮助我们解决在ETL方面遇到的可扩展性挑战，但是我们很高兴拥有一个能够正常运行的包含几百万兆(T)数据的数据仓库。

然后，我们打开点击流日志：仅仅第一天，在Oracle数据库上就发送了400GB的无结构化数据。我们再次对使用数据仓库表示怀疑。

超越数据仓库

根据互联网数据中心(Iternet Data Center,IDC)显示，数字世界在2011年将超过1800exabyte（1exabyte=2^{16}byte)。这么庞大的数据将无法通过关系数据库来管理。因此，对数据库管理系统存在很紧迫的需求，要求该系统能够从无结构化数据和结构化数据中抽取信息，但是人们对如何进展没有达成一致。

特别是，自然语言数据量丰富，信息量大，但是数据仓库管理不善。自然语言和其他无结构化数据通常在文档库和语音记录中获取，为了管理这些数据，企业组织开始超越了数据仓库供应商的产品，探索很多新的领域，包括企业搜索。

虽然很多搜索公司构建了很多工具来收集很多超链接文档即万维网(WW)，一些企业搜索公司选择重点研究内部文档集的管理。Autonomy公司成立于1996年，由一些剑桥大学的研究人员组成，他们充分利用贝叶斯推导算法(Byesian inference algorithms)来帮助定位重要的文档。Fast Search and Transfer(FAST)公司于1997年在挪威成立，其核心技术是更直接的关键字搜索和排序。两年后，Endeca公司成立了，其核心是研究使用结构化的元数据遍历文档集，该技术即“分面搜索”(fcted search)技术。G公司看到了其在搜索领域专长的机遇，在2000年推出了企业搜索设施。

在短短的几年内，企业搜索已经成长为拥有几十亿美元的市场，该

市场和数据仓库市场几乎完全分离。Endeca拥有一些处理更传统的商业智能的工具，一些数据库供应商为系统引入了文本挖掘能力，但是对结构化的和无结构化的企业数据进行管理仍然尚未实现一个完善的、集成的解决方案。

企业搜索和数据仓库都是技术解决方案，这些方案是为了最大化利用企业的信息资源来改进性能。早在1944年，MIT教授Kurt Lewin提出了“行为研究”(ation research)的螺旋式框架，其每个螺旋阶段是由计划、行为以及关于行为结果的事实发现组成的循环。对该问题更现代的方法是Peter Senge的“学习型组织”(Larning Organization)概念，其思想在他的书《The fifth Discipline》(Boadway Business出版）中做了详细阐述。这两种管理理论在很大程度上都依赖于企业组织根据先前收集到的行为信息做出反思并适应的能力。从这个角度来说，信息平台是一个学习型组织用于摄取、处理和生成实现螺旋式行为研究的必要信息的基础设施。

讨论完结构化和无结构化的数据管理，让我们一起回到Facebook的故事。

“猎豹”和“大象”[1]

从第一天开始对Facebook的点击流写日志起，到现在我们已经收集了超过400GB的数据。对该数据集的加载、索引和聚集操作对Oracle数据库的负载很重。虽然做了很多优化操作，但是我们还是无法在24小时内完成对一天的点击流的聚集操作。很显然，我们需要把日志文件聚集到数据库外，只在数据库中保存摘要信息(smmary information)供后期查询。

幸运的是，一个来自某大型网站的顶尖工程师加入了我们团队，他有过处理大规模Web点击流的经验。仅仅几周的时间，该工程师就构建了一个名为Cheetah（猎豹）的并发日志处理系统，该系统能够在两个小时内处理一天的点击流。这实在太让人振奋了。

但是，Cheetah存在一些不足：首先，在处理完点击流数据后，原始数据还是以归档方式保存(achival storage)，不能够被再次查询。此外，Cheetah是从一个共享的NetApp归档数据中获取点击流数据，而NetApp归档数据的读带宽受限。每个日志文件的“模式”(shema)是嵌入在处理脚本中，而不是保存为可查询格式。我们没有收集进程信息，而是通过Unix基础工具cron来调用Cheetah任务，因此无法应用复杂的加载共享逻辑。最重要的是，Cheetah不是开源的。我们团队很小，资源有限，无法分配更多的资源来开发、维护和给新用户培训使用Cheetah系统。

Apache的Hadoop项目，由Doug Cutting和Mike Cafarella于2005年末

启动，是我们取代Cheetah的最佳选择。以Doug的孩子的玩具大象命名，Hadoop项目的目标是实现遵从Apache 2.0许可的G公司的分布式文件系统和MapReduce技术。雅虎在2006年1月聘用了Doug Cutting，并投入了大量的工程资源来开发Hadoop。在2006年4月，该软件使用188台服务器，能够在47小时内，对1.9T的数据进行排序。虽然Hadoop的设计在很多方面优于Cheetah，但它在那时还太慢了，不能够满足我们的需求。在2008年4月，Hadoop用910台服务器，可以在209秒内对1T的数据进行排序。由于Hadoop性能的改进，我说服了运行组团队利用60台Web服务器和3台500GB的SATA驱动器，开始在Facebook第一次部署Hadoop集群。

在最开始，我们通过流方式在Hadoop和Cheetah中都导入一部分日志。Hadoop增强的编程能力加上其能够查询历史数据，从而推动了一些其他有趣的项目。其中一个应用是对所有Facebook用户交互的有向对进行打分来确定这些用户的亲密程度；这个分数可以被用于搜索和新闻订阅的排序。过了一段时间，我们把所有的Cheetah工作流都迁移到Hadoop上，废弃了前者。后来，事务数据库收集程序也都迁移到了Hadoop。

有了Hadoop,Facebook的基础设施可以支持对无结构化和结构化的数据的大规模分析。随着平台扩展为每天几百TB的数据规模，可以执行成千上万个任务，我们发现由于现在系统能够存储和检索的数据规模很大，我们可以构建新的应用，探索新问题的答案。

当Facebook向所有的用户开放注册，用户数在一些国家增长迅猛。但是在那时，我们无法根据国家执行点击流粒度分析。自从有了Hadoop集群，我们可以通过加载所有的历史访问日志到Hadoop，写一些简单的MapReduce任务来重新分析Facebook在一些国家，如加拿大和挪威，增长迅猛的原因。

Facebook的用户每天都有几百万的半公开(smi-public)的对话。据一次内部估算，留言板的数据量是博客的10倍！但是，这些对话的内容还是无法进行访问用来数据分析。在2007年，一个对语言学和统计学有强烈兴趣的暑期实习生Roddy Lindsay加入了数据组。通过Hadoop,Roddy能够独立构建一个强大的趋势分析系统，该系统名为Lexicon，每天晚上能够处理TB级别的留言板数据。可以通过URL:http：//facebook.com/lexicon查看结果。

在为Facebook应用构建信誉积分系统时，我们证明了把不同系统的数据存储到相同的存储库中会导致严重的问题。在2007年5月启动了Facebook平台后不久，我们的用户就被“淹没”在添加应用的请求中。我们很快意识到需要添加一个工具来识别有用的应用和用户认为是spam（垃圾）的应用。通过收集API服务器的数据、用户信息以及来自网站本身的行为数据，系统能够构建一个模型对应用进行打分，这使得系统可以分发我们认为对用户最有用的应用邀请。

[1]猎豹和大象在此采用了借代的修辞方法。猎豹(ceetah)指的是Facebook的The Cheetah日志处理系统，大象(eephant)指代的是Hadoop项

目，具体参见下文。

不合理的数据有效性

在最近的文章中，G公司研究员发表了三部曲，提炼出他们在尝试解决机器学习中的一些最困难的挑战的心得。当讨论到语音识别和机器翻译，他们认为“简单的模型和大量的数据总是胜过在少量数据上构建的复杂(eaborate)模型。”我不想和他们的发现争论，但是，当然存在某些领域，更复杂的模型会更成功。但是基于G公司研究员的经验，确实存在大量问题，更多的数据和更简单的模型对它们会更有效。

在Facebook,Hadoop是我们探索不合理的数据有效性的工具。比如，当把Facebook网站翻译为其他语言时，我们试着征集那些母语为特定语言的用户，以便帮助我们完成翻译任务。我们的一个数据科学家Cameron Marlow，对所有的Wikipedia进行了爬虫，对每种语言构建了特征三元频率计数(caracter trigram frequency counts)。使用这些频率计数，他构建了一个简单的分类器，该分类器可以通过识别用户的一组留言来确定他的母语。使用该分类器，我们可以通过有针对性方式，积极地征集用户加入我们的翻译项目中。Facebook和G公司在很多应用中都用自然语言数据，详情请参看本书第14章Peter Norvig对于该课题的探讨。

G公司的观点指出了对现代商业智能系统的第三条变革：除了在一个系统中管理结构化和无结构化的数据，这些商业智能系统必须能够扩展到可以存储足够多的数据，使得可以采取“简单模型，大量数据”的方

法来实践机器学习。

新工具和应用研究

在Facebook，绝大部分Hadoop集群的早期用户都是渴望追求新兴技术的工程师。为了使企业的更多人可以访问信息，我们在Hadoop上构建了一个数据仓库框架，并称为Hive。Hive的查询语言类似于SQL，支持嵌入MapReduce逻辑、表分区、抽样和处理任意序列化数据的能力。最后一个特征至关重要，因为收集到Hadoop的数据在结构上不断变化；允许用户指定自己的序列化模式，可以使我们把为数据指定结构问题转为把数据加载到Hive。此外，我们还实现了一个简单的用户界面(uer interface,UI)来构建Hive查询，名为Hipal。使用这些新的工具，市场、产品管理、销售和客户服务的非工程师都能够在几TB的数据上自己执行查询。经过几个月的内部使用后，在Apache 2.0许可下，Hive成为Hadoop的官方子系统，现在仍然在积极地开发中。

除了Hive，我们构建了分享图表和图形的门户Argus（受IBM的Many Eyes项目启发）、工作流管理系统Databee、用Python写MapReduce脚本的框架PyHive、为终端用户提供结构化数据服务的存储系统Cassandra（现在作为开源，在Apache孵化器中）。

随着这些新系统的稳定，我们最终构建了由单一Hadoop集群管理的多层模式(mliple tiers)的数据。企业中的所有数据，包括应用日志、事务数据库和Web爬虫，都以原始数据格式，定期收集到Hadoop分布式文件系统中(HFS)。夜间执行的几万个Databee进程将把一部分数据转化为结

构化格式，把它放入由Hive管理的HDFS文件目录中。在Hive中执行下一步聚集操作，用来生成Argus服务报表。此外，在HDFS内，在自己的home目录下维护“沙盒”(sndboxes)的工程师可以运行原型(pototype)任务。

目前，Hadoop包含了将近25亿兆（2.5PB)的数据，而且以每天15TB的数量级增加。每天都有3000个以上的MapReduce任务在运行，处理55TB的数据。为了适应这些运行在集群上的任务的不同优先级，我们构建了作业调度器，实现在多个队列上的资源共享。除了支持内部和外部的报表、a/b测试管道和很多不同的数据密集型产品和服务，Facebook的Hadoop集群可以实现一些有趣的应用研究项目。

由数据科学家Itamar Rosenn和Cameron Marlow主持的一个纵向研究项目(lngitudinal study)用于预测长期的用户参与的最重要的因素是什么。我们使用信息平台来选择一些用户的样本，删除游离点，并对参与度的不同尺度使用一些最小角度回归技术(last-angle regression)来生成大量的特性。有些特性能够通过Hadoop生成，包含计算好友网络密度的各种尺度和基于信息特性的用户范围。

另一个探索激励新用户贡献内容的动机的内部研究，在2009年CHI会议的论文“Feed Me:Motivating Newcomer Contribution in Social Network Sites”中描述。Facebook数据组的一个更新的研究是查看信息流是如何在Facebook的社会图中流动，该研究的标题为“Gesundheit!Modeling Contagion through Facebook News Feed”，已被

2009 ICWSM会议接收。

在Facebook，每天收集证据、测试假设、构建应用和使用共享的信息平台生成新的洞察。而在Facebook之外，其他公司也同时构建了类似的系统。

MAD技术和Cosmos

2009年VLDB会议的“MAD Skills:New Analysis Practices for Big Data”一文详细描述了Fox Interactive Media(FIM)公司的分析环境。结合Hadoop和Greenplum数据库系统，该团队在FIM中构建了我们很熟悉的数据处理平台，但其工作和我们在Facebook的工作是独立的。

该文章的标题谈及了FIM平台的三个方面：磁性(Mgnetic)、灵活(Aile)和有深度(Dep)。“磁性”表示存储该企业的所有数据，而不仅仅是适合企业数据模型的结构化数据。同样的思路，一个“灵活”的平台需要能够优雅地处理模式变化，使得分析员能够直接在数据上工作，或者根据需求对数据模型不断演化。“有深度”表示在数据上执行更复杂的统计分析实践。

在FIM环境中，在单Greenplum数据库内，数据分为呈现(saging)、生产、报表和沙箱四种模式，和之前描述的在Facebook中Hadoop内的多层模式很相似。

微软单独发表了其数据管理栈(dta management stack)的细节。在标题为“Dryad:Distributed Sata-Parallel Programs from Sequential Building Blocks”和“SCOPE:Easy and Efficient Parallel Processing of Massive Data Sets”两篇文章中，微软描述了和我们在Facebook构建的极为相似的信息平台。它的基础设施包含分布式文件系统Cosmos和并行数据处理系统Dryad，它还发明了类似SQL的查询语言SCOPE。

三个团队在完全独立的技术团队工作，却开发演化了处理大规模数据的相似平台。这是怎么回事呢？通过把特定结构的需求与存储数据的能力以及为数据检索创造API进行分离，大规模网站的存储系统看起来更像数据空间(dtaspace)而不是数据库。

作为数据空间的信息平台

在雅虎、Quantcast和Last.fm公司，也有类似的几十亿兆规模(ptabyte-scale)的数据平台。这些平台不完全是数据仓库，因为它们通常不使用关系数据库或者任何数据仓库建模技术。它们也并不完全是企业搜索系统，因为只对一部分的数据构建索引，而且它们对外开放了更丰富的API。除了传统的数据分析工作，它们通常用于构建产品和服务。

与大脑和图书馆类似，这些共享的数据处理平台为公司摄取、处理和生成信息提供了场所，幸运的话，它们还可以加速企业从经验数据中学习的步伐。

在数据库社区，研究议程已经开始从纯关系数据管理过渡到在大数据集上存储和查询的更经典的系统，称为“数据空间”。在论文“From Databases to Dataspaces:A New Abstraction for Information Management”(htp：//www.eecs.berkeley.edu/～franklin/Papers/dataspaceSR.pdf)中，作者强调了存储系统必须支持所有数据格式，并提供一些API供数据访问，这些API是基于存储系统对数据的理解不断演化的。

我们之前描述的信息平台是数据空间的真实例子：用单一存储系统管理企业中各个部分的所有的结构化和无结构化的几十亿兆的数据，该系统需要给工程师、分析员和报告员提供各种数据访问API。由于在工业上这些系统大量存在，我期待数据库社区继续探索数据空间的理论基

础和实践意义。

信息平台是构建学习型组织的基础设施的关键部分。人们在加速学习和利用信息平台过程中，开始扮演了数据科学家(te Data Scientist)的角色。

数据科学家

在最近的访谈中，G公司首席经济学家Hal Varian强调了员工需要能够从之前描述的信息平台中抽取信息。正如Varian所言：“找到能够为一些变得普遍且廉价的东西提供稀缺、互补的服务。那么，是什么变得普遍且廉价？数据。是什么与数据相辅相成？分析。”

在Facebook，我们发现传统的头衔如商业分析师、统计学家、工程师和研究科学家都不能确切地定义我们团队的角色。该角色的工作是变化多样的：在任意给定的一天，团队的一个成员可以用Python实现一个多阶段的处理管道流、设计假设检验、用工具R在数据样本上执行回归测试、在Hadoop上为数据密集型产品或服务设计和实现算法，或者把我们分析的结果以清晰简洁的方式展示给企业的其他成员。为了掌握完成这多方面任务需要的技术，我们创造了“数据科学家”这种角色。

在金融服务领域已经构建了历史市场行为的大数据存储作为该领域的数据科学家，即数据分析专家(Qants)，来开发新模型的实验场。在工业以外，我发现在很多科学领域，研究生扮演着数据科学家的角色。Facebook数据组团队的其中一员曾在生物信息实验室工作过，在那里他构建过数据管道流，并做类似的离线数据分析。在CERN，著名的Large Hadron Collider生成大量的数据，这些数据是由一群追求突破的研究生精心收集和钻研的。

最近新出的书如Davenport和Harris合著的《Competing on Analytics》

（哈佛商学院出版社，2007），Baker的《The Numerati》(Hughton Mifflin Harcourt，2008）以及Ayres的《Super Crunchers》(Bntam，2008）都强调了在跨工业中数据科学家的重要性，他们在促进企业基于收集到的信息做出改进发挥了至关重要的作用。和研究社区在数据空间的调研一起，数据科学家在今后几年需要进一步的定义。通过更好的阐明数据科学家角色，我们可以建设培训课程、制定广告层次、组织会议、写书以及为任何被认可的行业做补充。在这个过程中，可行的数据科学家组织将会不断扩展，用来满足飞速增殖的数据平台上不断增长的专业“领航员”需求，进一步加速跨企业的学习过程。

结论

当面对在Facebook构建一个信息平台的挑战时，我发现观察别人是如何跨越时间和问题领域来解决相同的问题是很有帮助的。作为工程师，我最初的做法是通过已有可得的技术作为指导，这在现在看来显得有点目光短浅。最大的挑战是一直致力于研究构建“学习型组织”的基础平台和人员构成(hman component)这个大的问题，而不是某些特定的技术系统，如数据仓库或企业搜索系统。

我确信构建信息平台采用的硬件和软件将会迅速演化，并且数据科学家需要掌握的技术也将以同样的速度变化。保持致力于加速学习过程的目标对于企业组织和科学都有帮助。未来属于数据科学家！

第6章　照片档案的地理之美

Jason Dykes和JoWood

照片可以是美丽的。把能够捕捉精彩瞬间、点燃激情并能激发崇高的东西仅仅看做是数据，这是对照片的一种贬低。然而一旦以数字形式存储，处理相片的二进制数据可以与处理任何其他的数值型数据流一样。但是我们可以对这些照片做进一步处理：收集在一起、排列整理、添加描述标注，由此我们可以创建出一个场景，并且一个新的美好的东西也由此而生。这种美虽源于组成它的所有照片，但却又远远超过了每张照片的所有美丽汇总。

在本章中，我们将探索以地理的视角审视一组照片时，所能发现的美。我们将通过地图和其他图形来对地理特性进行可视化检查。这里的“地理”指的是允许我们把一些东西同某个地方(pace)或位置(lcation)（这是两个截然不同的概念）关联在一起的信息。当我们处理数据时，有很多相关的地理信息。有些人估计大约有80%的数据是地理数据(McEachren和Kraak 2001）。地理信息可能是直接以经纬度坐标的形式进行记录，或者是这个地方的邮政编码、名称或一些其他的信息。这种地理数据将是组织、过滤和解释数据的有用的方式。记录不断增长的、巨大的数据集中的地理信息，可能会为关于某个地方的观点分析提供格外有用的信息资源。

可以通过很多途径把地理信息和数据进行关联。这个途径可能是数

据收集过程的组成部分（例如，卫星遥感）；也可能产生于查询和解释数据的过程中（例如，以G公司本地搜索为代表的一些地理位置相关的服务）；也可能产生于更先进复杂的时空分析过程中，它是“直觉构建”(Snsemaking)过程的一部分，且当前的地理可视化分析正是其典型代表(Adrienko等2008）。在这里，我们采用在收集过程中已经通过特定的定位方式取得了其地理信息的数据作为起始点数据，但这些起始点数据也包含了一些额外的、地点不明晰的描述。地理档案包含了超过100万的带有精确的经纬度的照片。这些照片的经纬度有的是通过带GPS功能的设备如iPhone获得的，有的则是由一些人为的方式在地图上手工定位了他们的照片。此外，这些照片的地理信息也有可能是它们的拥有者以非格式化的文本形式提供描述，或者可能是以邻近地区的名字给照片命名，或者是对照片捕捉到的特征或者活动进行描述。这其中有一定的复杂性和精妙性，而且我们应该明白，当我们试图通过可视化方式来加强对位置和地点描述之间相互作用的知识时，美丽将“油然而生”。

当我们编写优美的代码时，通常情况下都有很具体的目的，比如对列表进行排序，求解线性方程组，或者完成傅里叶变换。代码之美源于达成目的(Klawa 2007）。当处理美丽的数据时，我们的目标往往不是非常明确。分析数据是科学探索的一个重要环节，而且可以带给人们更深的洞察力，并且测试假设情况和验证先前的理论。美丽的数据值得探索。它包含的模式、结构和异常情况虽然不能立即显现出来，但是在进行更深的深度挖掘时，它就显现出来。我们在工作中通过可视化方式探

索数据过程中，形成了两个长期的传统。为了进行信息通信以及支持知识发现，制图学已经发展了强大的技术来通过可视化方式表示地理数据。几个世纪以来，它已经成功地将开拓创新、科学活力和更多主观的设计和批判技巧相结合。地图本身可以与它们所描述的对象同样美丽。信息可视化涵盖了数据的可视化探索过程，该过程可能与地理信息毫不相关，而只是通过设计和图形、图表以及相关的交互行为。在本章中，我们将报告一些融合了制图学和信息可视化的基础原理的探索地理数据之美的方法。

数据之美：Geograph项目

我们认为Geograph项目[1]是优美的数据集合，这里有很多方面的原因。描述不列颠群岛(Bitish Isles)的地理的地理对象和标注照片的档案正在快速增长，它是有价值的数据源、在线社区、游戏和乡村探索驱动的完美结合，无论是宏观美还是细节美，都可以让人产生无限遐想。

Geograph项目最初由Gary Rogers提出，后来由Paul Dixon、Barry Hunter和不断发展壮大的各个版面的版主团队提供支持，它旨在收集英国和爱尔兰的每一平方公里上的“具有代表性的图片和信息”(Gograph 2009）。到本文写作为止，该项目已经有超过8500个贡献者，共收集记录了120多万张图片，涵盖了英国244000平方公里国土面积的90%以上。图6-1给出了一个例子。该数字无疑会随着时间不断增长——当你正阅读本章时，英国的乡村正在“地理化”。



图　6-1：Aberuchill附近的小路。远处是Bioran　　Dalchonzie。这是Geograph项目的网站出现的第100万张图片(htp：//www.geograph.org.uk/photo/1006884）。图片版权归Dr.Richard Murray所有，由Creative　　Commons License(http：//creativecommons.org/licenses/by-sa/2.0/）授权使用（见彩图12）

Geograph项目的参与贡献者可以自由地选择他们的典型风景，但是地理特征应该能够表达1km的网格范围内的典型的人文和物质地理特征：

想象一下，在地理课上，一个小孩希望弄清楚一个特定的网格范围内的人文和物质地理特征实际上看起来是什么样子，当他看着地图时，他可能会找到对他有帮助的东西(Hwgood等2007）。

地理的很多方面蕴含美：

创造一个有目的地选择并标注的开放图像库的目标是简单性（想法）、复杂性（处·理，其过程也是包含着几丝美丽）、实用性（资源）的“动人”组合。为了维护该集合并使得它可用，推动该项目所付出的组织工作和努力是令人钦佩和印象深刻的。

·通过“自下而上”(bttom-up)、“集体力量”(cllective effort)的方式生成档案，以及使用技术和个人力量来实现这些目标，这些方式中无不蕴涵着美丽。在协作式的“以人作为传感器采集信息”(ctizens as sensors)(Godchild 2007），该项目本质上的集体之美，是基于不考虑个人利益的共同理解和广泛合作，这一点值得一提。

·通过引人入胜、易于理解和激动人心的Web站点来实现和展现该想法，该网站还提供了很多种方式来访问信息，这些也体现了项目在艺术和技术上的品质。

根据地理信息生成的不同尺度的地图，提供了生成该档案集合的各种过程的洞察·力，这些也蕴涵着艺术品质（见图6-2）。这些包含很多富于创造性的制图表示和提供访问档案集合的大量信息和更新的交互特征，比如参与贡献者以及其贡献的照片的地理密度和分布地图。

个人贡献的照片作为人文和物质景观的表示，是由那些居住和游览图片所描述的地·方的人们所决定的，这些照片具有艺术魅力。Geograph项目社区将这种魅力“正式化”，因为“地理年度”(te geograph of the year,GOTY)的候选图片是通过每周从贡献的图片中选出来的。

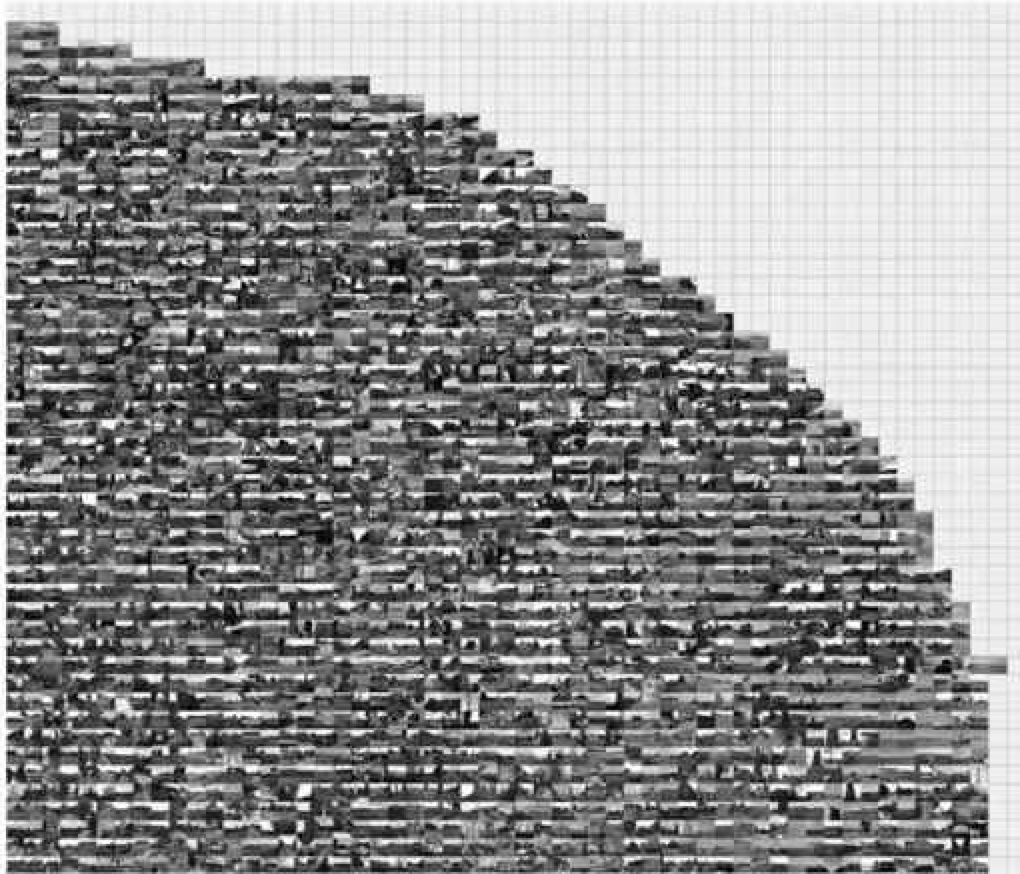


图　6-2：Norfolk海岸的地理马赛克图。每个图片都是根据其地理位置进行映射，表示了1平方公里的景观。图片由Creative　　　　Commons License(http：//creativecommons.org/licenses/by-sa/2.0）授权使用（见彩图13）

Geograph项目在某种程度上是典型的广泛的用户生成的描述地理的数据集合，这些数据正变得可以访问，而这种种特征使得Geograph项目格外显著。存在很多真正的机遇来探索Geograph项目的数据，并通过这些数据孕育新的知识，这些知识可能会进一步加强Geograph数据之美。我们在本章的后面部分将举出一些例子，用来描述在档案里集中展现一些地理视觉探索。

[1]关于该项目的更多信息，可参考其Web站点：http：//www.geograph.org.uk/。

可视化、美丽和树形图

在我们深入探索Geograph项目档案本身之前，一些有助于探索档案集的动机和可视化技术是值得考虑的。传统制图绝大多数集中于可以通过传统媒体进行再制造的静态产品，这一点是可以理解的。在过去15年中，我们的绝大多数工作是交互式的——吸引数字技术来重新考虑地图的本质和角色，并且把它们作为可响应的图像方式，目的是为探索而查询(Fsher 1998）。我们的目标是保证在交互平滑性过程中的艺术品质，以及通过开发富信息的景象和激发思考和探索的动态行为所获得的满足。然而，我们最近的一些工作已经重新重视数据密度，重新集中于最根本的制图设计决策，这些决策和那些高效利用空间生成布局和符号学（数据编码）相关。制图工作的一部分是由硬件进步所驱动，这些进步使得处理和显示大的数据密集型图片变得更可行。实际上，该工作响应了以新的有效、优雅的方式来图形化表示各种更大的数据集合的需求；由于我们遵从了Tufte的建议“在小空间展现很多数字”(Tfte 1983），因而这些数据变得更易于访问。

可视化数据探索之美在哪里

我们把美丽看做和一些激励相关的主观品质，该品质可以带来积极的感性体验。在可视化过程中，通常都是开发者或设计者觉得美丽。为了更全面地审视艺术（如Kosara 2007），适当地号召项目社区正式地对可视化作品提出批评意见。但是，在一个可用的知识体系开发出来之

前，我们在开发优雅的图形上还是依赖于广泛的原则和经验法则。以上这些很多方面都已经应用于《Beautiful Code》(Oam和Wilson 2007），而且有助于数据可视化应用。例如，Brian Kernighan(Kernighan 2007）指出优雅的代码的特征，包括紧凑、优雅、高效和实用，而且通过“理想情况下，代码可以在单个页面显示”来非正式地量化紧凑性。Yukihiro Matsumoto认为让人难以理解的代码是不优雅的，而且应用该标准来开发Ruby编程语言(Mtsumoto 2007）。但是“难以理解”和复杂是不同的，不该混淆在一起。根据我们的评判标准，一个易于理解的简单图形但表示的数据信息量非常少谈不上优雅。相反地，美丽的数据可视化是通过易于理解的方式来表示复杂的东西——可能通过重点研究该数据的某几个方面或者突出某些特殊方面。这可能和Kernighan的原则一致：努力达到在单个“页面”上完成（数据可视化）的目标。我们的情况是通过以下几个方面来达到这个目标：努力扩展和综合已有的制图和信息可视化方法，从而高效地利用空间显示多个图片（在空间和其他方面的）关系；力求通过足够紧凑的方式，使得在一个页面或者屏幕可以完全显示，而且可以很优雅地同时展示全局结构(Gstalt)和局部细节（即时需求的细节）；基于任何特定的数据集上的需求，努力设计和开发易于理解和有用的图形，使得真正的用户能够理解和使用图形来解决已知的信息需求。

Tufte（1983）提出了“数据/墨水”比率的想法——这是一个启发式算法，鼓励平面设计师评估一个直接用于表示数据的页面的墨水比例。该

比例越高，使用图形符号化就越高效，而且其表示的信息就越有深度。这种形式和功能指标可能有助于数据图像之美。相似地，我们可以考虑数据/地理位置比率，即一个图形元素在一个页面的位置程度反映了它所表示的数据的特征。传统的地形图在这方面表现很好，因为一个符号在页面上的位置通常可以确定其所代表的地理位置。在这方面，很多信息图形可能不是很高效，如一些地图（比如统计地图和电路图）。我们认为空间的有效利用是优雅的数据可视化的重要方面，因为它支持地理（或者其他）模式的可视化发现过程。在包含地理特征的大量的数据集情况下，比如大量的志愿者贡献的数据集合，空间的有效利用就变得更加切实重要。简而言之，在高效使用和展现空间上蕴涵着美——尤其是为了展现地理。

使树形图变美：从地理角度出发

树形图是层次的填充空间式(sace-filling)展现方式(Sneiderman 1992），如图6-3所示。正如很多美丽的想法一样，树形图基于一个优雅简单的思想。把一个数据项表示成一个矩形。如果该数据项本身包含一些其他数据项（任意层次的特征定义）的集合，这些数据项的每个都是通过一个更小的矩形来表示，这些小矩形落在“父”矩形的内部。反过来说，这些小矩形本身也可以包含更小的“子”矩形等。这些矩形在排列上可以填充整个图形空间，而没有任何缝隙。每个矩形或节点可以根据其所表示的数据的一些特征来调整大小；还可以根据数据对它进行着色或者有意义地标注。所有的矩形都是可见的，它们不会交叠。这些矩形

在表示的紧凑性上具有一定的优雅性（单个着色和标注的矩形可以同时显示某些数据的三个或者更多独立的特征）。每个节点（一个矩形）的简单的几何图形本身就表示大量的数据集，因为一个树形图可以同时显示的节点数目几乎和一个屏幕上的所有像素一样多。由于层次的语义包含关系是直接表示为树形图中的几何包含关系（父节点包含子节点），这种方式让人感觉更优雅。

我们认为采用树形图来表示大量的地理和专题分类记录信息具备潜在可能性，而且我们发现通过构建新的分类这种方式探索Geograph数据集中的海量记录数是可能的。

然而，由于很多原因，树形图广受人们的批判。滑稽的是，人们批判的是树形图的艺术性(Cwthon和vande Moere 2007），但是我们认为这是功能实现上的问题，而不是设计本身。

更值得注意的是，在树形图中任意放置节点可以显著降低“数据/位置”的比例。绝大多数现有的树形图布局算法定位这些节点是为了最大化它们的“高/宽”的比例（使长方形尽可能接近方正，对于审美和大小比较任务是很重要的），而且为了提高可阅读性（最大化水平上的线性连续性）。很少有人会关心如何使用图形的位置来表示该数据的某些方面特征。因此，树形图包含线性不一致性以及节点位置的任意放置（见图6-3）。这些对立面创建了制图学和统计图形的最佳实践，其中平面上的位置被认为是表示关系信息的最主要方式(Brtin 1983）。在树形图内的节点的任意位置都无法利用“认知地理第一定律”(Fbrikant等2002），

该定律认为不同物体之间距离越近，就越相似。

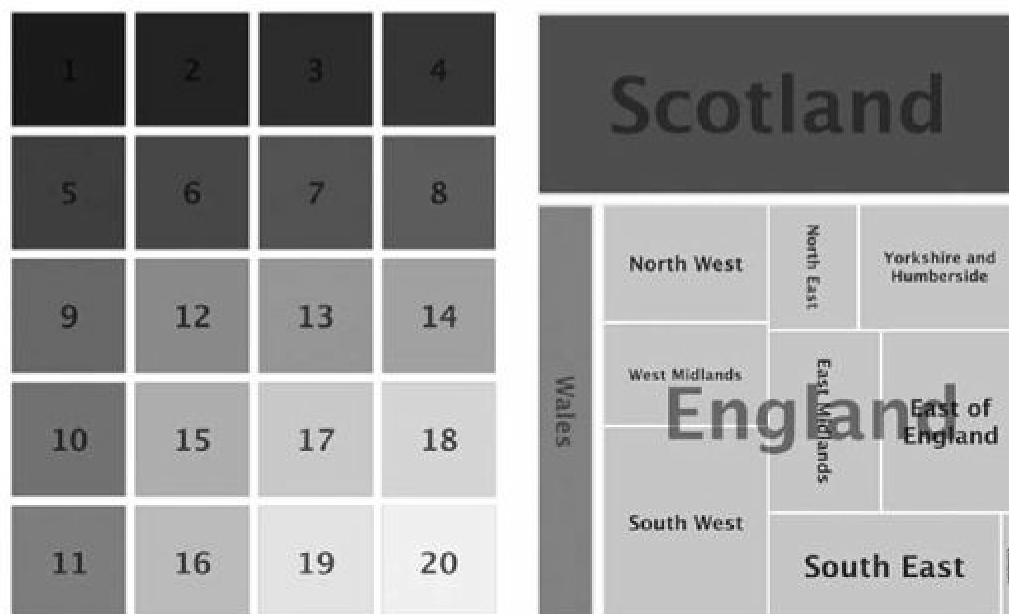


图　6-3：两个简单的树形图。（左图）通过传统的方块化(suarified)布局方式（按序着色）放置的20个有序节点。注意没有使用前后一致的位置来表示有序的1～20序列。（右图）空间树形图，其节点是根据它们相应的地理位置（按范围着色）来放置的

我们认为在一个二维树形图中，根据数据中的一维次序或者二维次序，对所有层次的节点进行排序是可行的(Wod和Dykes 2008）。通过这种处理方式，我们解决了和树形图关联的一个关键问题，即主要的信息载体维度，通过一维（或者多维）的数据维度对它们进行映射，但这并没有达到充分利用的效果。简而言之，我们在树形图内部使用空间来表示一维上有序的或者二维上按空间排列的数据。

Geograph在使用条款上的观点

“地方”(pace)的概念是很复杂的，不能简单地通过经纬度来很好地描述。地方不仅仅是一个位置，而且它也说明了创建一个地方的感觉所包含的特征本质。它可以依赖于无形的、主观的甚至有时对立的特征，这些特征传统上无法通过数字数据集来很好地进行展示。志愿者收集的或社区参与者贡献的地理信息，如可以通过Geograph项目获取的地方的个人描述，为我们提供了新的多角度方式来访问信息。这些可能会反映很多观点，而且当我们努力通过更有效的方式来描述一个地方时，促使我们开始考虑对该地方的其他想法。

Ross Purves和Alistair Edwardes在Zurich大学做研究时，一直使用Geograph项目的数据作为描述地方的参考来源。他们的最终目标是通过为描述地理的数字图像自动添加索引项，用来改进信息检索，这些数字图形和某个地方的流行观点相关，如“山脉”、“偏远”或“徒步旅行”。他们的工作涉及验证先前的研究，并通过比较Geograph项目的数据和当前为了描述地方所付出的努力来形成新的观点，以及分析在地理描述中的术语共现度(Ewardes和Purves 2007）。

在学术上流行的方法涉及识别所描述地方的基层或场景类型。这些对地方的综合描述已经通过传统地人为主观测试的方式推导出。这些描述之间很难协调，而且通常参与的人数很少，使得人们很难对结果进行泛化或者重新做那些实验。Edwardes和Purves评估了Geograph项目的参

与者所使用的方式，对场景类型如山地、丘陵、山谷、河流、岩石、湖泊、峡谷、悬崖、海洋、洞穴进行排序，发现和参与者的研究中报告的术语以及这些术语使用程度有很大关联(Ewardes和Purves 2007）。

随着数据集中所使用的术语在一定程度上得到了验证，我们共同确定探索自然、结构和Geograph项目数据三者之间的一些关系的地理特征存在着机遇。特别地，我们期望能够理解以下几个方面之间的关系：照片内容、照片位置以及这些照片提供者的标注中记录的关于某个地方的文本描述。可视化方式看起来很合适，而且树形图技术使我们能够探索Geograph项目的以上这些特征。

以下的例子记录了一些方式，这些方式中的空间树形图和其他图形被用于探索Geograph项目，因为我们开发了该项目数据集的共享知识库，用来表达我们对地方的描述的理解。

表示术语分类

Geograph项目档案在2008年4月进行了处理，当时大约为75万的图片提供了标题和文本描述来说明其地理信息。我们重点研究与六个基础层次相关的图片，或者通过Edwardes和Purves的分析，被认为特别有意思的场景：海滩、乡村、城市、公园、山脉和丘陵。对于这些场景类型，我们为每个在以下三个不同方面选择了最受欢迎的描述词：活动（主要是动词）、元素（主要是名词）和品质（主要是形容词）。这种方式产生了六种场景类型的术语共现分类，每种场景包括三个方面，每个方面包含和场景类型相关的很多描述符。树形图反映了对于选定的包含一个

流行描述符的场景类型，其每个共现度所包含的节点的分类，并展示了在Geograph项目所使用的描述的一些结构。图6-4所表示的树形图采用了有序的方块化布局算法来优化节点的形状和位置一致性(Wod和Dykes 2008）。叶子节点（各个子图）是单位大小。



图 6-4：对于六种选定的场景类型，其出现在地理标题和评论的描述的树形图。节点大小表示术语的出现频率。颜色突出了包含一种继承随机方案的场景类型/刻面(fcet)/描述符分类。布局采用“有序的方块化”方法来维持节点之间的方块形状（见彩图14）

图6-4中的每个节点都采用一种继承随机方案来对场景类型进行着色，而且孩子节点（刻面、描述符和地理科学本身）继承了这种着色方案，只有微小的颜色差别。虽然颜色本身并没有独立的涵义，但是采用

这种着色方案来突出这种分类方式的层次结构。这种布局和颜色编码的结合有助于我们通过展现结构和鼓励视觉对比来探索数据。例如，我们发现丘陵(hll)这个词比公园、乡村、城市、沙滩和山脉这些词更流行，因为它在树形图中占了更大的区域面积。在这些场景类型中，元素刻面始终比品质或活动更为流行。活动方面和公园相关性特别强。描述符“公路”在四种场景类型中都居首要地位，而不是“沙滩”或者“山脉”。对于丘陵和山脉，描述符如山谷和小路与丘陵一起使用得更频繁，虽然相对而言，这些描述和山脉更相关。而湖泊和山脉一起使用得更频繁。小径是对与丘陵相关的流行的描述符，而不是山脉。这种数据密集的信息之美的一个重要方面是很多其他关系是同时显示，而且可能是同样被报道。我们可以通过远远多于1000个的文字来继续描述这些图片。

基于对各种Geograph的图形数据的理解上的经验，我们这里已经采取了一系列的设计决策。一个有用的可选方案是采用“切片和切块”的布局算法。其结果是一个马赛克图，使得其内的每个场景类型和刻面的大小比例更易于关联在一起，因为是通过长度而不是区域来进行比较（在图6-5上方）。但是，把生成的节点拉长，意味着不同描述之间的标注和大小估计会变得更加困难。一个折中的方案是应用有序的方块化算法来对叶子节点进行排列（在图6-5底部）。通过这种在不同的分类级别所做的布局和颜色实验，有助于我们突出和探索数据集中的各种显著的“品质”。

road
farm
lane
track
footpath
valley
hill
(ACTIVITY)

car
road
house
entrance
park
(ACTIVITY)
country

road
church
lane
street
house
hall
houses
village
green
(QUALITY)

sea
bay
tide
sand
coast
cliffs
cliff
path

road
hall

road farm
lane track footpath
valley trees square junction
hill
(ELEMENT)
path forest wood summit
slopes castle fields
hills
fort
land
trig
iron
grazing

car road
house entrance trees
estate building lake
(ELEMENT)
wood hall path forest
park
town gate
tree
city
oak bus
country
walk

road church
lane street
house hall houses
(ELEMENT)
sign village pub
school post
well stone
inn
pond
green

sea
bay
tide
sand
car

图　6-5：对于六种选定的场景类型，其出现在地理标题和评论的描述的树形图。节点大小表示术语出现频率。颜色突出了包含一种继承的随机方案的场景类型/刻面/描述符分类。布局采用“切片和切块”的方法来增强维度（上图）间的比较和“切片和切块/有序的方块化”方法来改进标注的可读性，如下部的图（见彩图15）

通过颜色表示绝对地理位置

虽然图6-4和图6-5中的树形图提供了如何描述地方的一些信息，但是它们几乎没有提到该地方与地理位置之间的关系。我们探索了一系列的方式给树形图增加地理位置信息。第一种方式是采用颜色来提供大不列颠群岛之内的绝对地理位置信息。Geograph数据的地理位置信息是通过“东向”(esting)和“北向”(nrthing)坐标组的投影来存储的，它们记录了该地理位置到英国国家电网(Bitish National Grid)起源地区的东部和北部距离。

我们面临的挑战是通过一个颜色来表示照片地理位置的两个维度（东向和北向），该颜色能够把该照片和其他不同地区的照片区别开。绝大多数的颜色空间是通过三个部分（如红色、绿色和蓝色，或者色调、饱和度和亮度）定义的，因此只选择两个部分来表示一组坐标是有问题的。此外，绝大多数颜色方案在主观感觉上就是不一致的；换句话说，两种颜色之间在色彩板上存在固定的视觉差距。因此，我们决定采用CIELab颜色模型，其提供了一种在感官上感觉更一致的色阶方案。通过色彩空间的a和b两个分量表示每个照片的“东向”和“北向”位置，我们

能够生成一个颜色地图，其中西南地区是橙色的，东南地区是绿色的，东北地区是蓝色的，而西北地区是紫色的。中部地区趋向于褐色，两个节点间的颜色相似度说明了它们所表示的照片的地理位置的相似度。通过这种方式对图6-6有序的方块化树形图中的节点（和图6-4不同）进行着色。



图　6-6：“有序的方块化”树形图，其颜色是通过CIELab颜色模型的一种色彩空间来显示绝对的地理位置，在该空间中，主观感觉到的颜色区别和地理位置的差异紧密相关（见彩图16）

从这个观点来看，地理位置对地方的描述带来的影响是显而易见的，这一点可能鼓励我们从空间角度来探索Geograph项目的数据档案。例如，小径、顶峰和石冢展示了高山草地不同的地理特征；白垩、古冢

和交界点则表示丘陵的不同地理特征；而山脉、沙滩、乡村有不同的地理位置特征；在沙滩内的活动和品质有显著不同的地理特征。地理位置和地方之间相关的一些复杂性也很明显。

通过空间树形图表示相对地理位置

使用颜色来表示地理位置有一定的艺术之美，并为地理位置-地方(lcation-place)的关系提供了一定的洞察力，但是其在有效性上具有一定的局限性。特别地，它要求用户为颜色和地理位置的对应关系分配内存。图6-6显示的树形图也没有以任何有意义的方式来使用节点位置。因此，我们可以把每个照片的地理位置映射到树形图中的节点位置，这样北部的照片会出现在每个封闭的节点空间的上方，西部的照片出现在左上方等。由于树形图会通过非交叠矩形来填充空间，我们无法提供精确的地理位置空间映射，但是这种布局方式确实说明了节点的相对位置，因而增加了“数据/地理位置”的比率。如果我们对探索地理位置方面的地方描述符感兴趣，则可以采用CIELab着色方案来突出绝对的地理位置，或者在保留很强的制图隐式表现前提下，用颜色表示数据的一些其他方面（比如术语的重要性）。图6-7保留了有序的方块化布局方式来进行术语分类，但是根据地理位置，对每个描述符的节点进行重新排列。

在该图中，山脉的粉色、紫色和褐色显示了该术语是用于北部和西部，虽然如围栏、垫木、凹地和黑色这类术语和这种模式不一致，但是它们也显示了自己的地理特征。沙滩的生动形象的色彩显示了这种场景

类型周围的海岸的自然特性，而城市更柔和的颜色则表示它是中央地带。

图6-8通过对所有节点使用空间排序方法，更进一步加强了显示效果。它显示了选定的六种场景类型的地理和分类特征。

表示地理位置位移

虽然在分类结构中，我们可以识别出一些空间模式，而为了使节点能够完全镶嵌于树形图空间中，节点的显示和真实的地理位置的偏移程度是不确定的。CIELab的着色方案可以提供这种偏移的表示；注意图6-7中的沙滩的品质和活动方面的不同颜色，或者在城市/元素/舞厅和小山/元素/农场之间的不一致性。我们可以做出进一步的改变，但是这种改变是通过说明在地理上，一张照片或者一组照片如何在镶嵌过程中从地理坐标转换过来。为了实现这点，我们遵循Skupin和Fabrikant（2003）的建议，他们认为认知上似是而非的制图需要采用合适的方法来表达这种位置错误。



图 6-7：“有序的方块化”树形图，通过CIELab色彩空间的颜色显示绝对的地理位置。包含描述符节点的叶子节点，使用空间有序算法，通过地理位置做关联排序（见彩图17）

图6-8在树形图上叠加了一组线条集合。这些线条把每个节点的树形图位置连接到其地理位置-线条越长，位移越大。例如，海滩上与品质和活动方面相关的位移向量，尽管它们在空间树形图中是并列的，但是这些位移向量证实了我们之前指出的不同地理特征。创建这些线条背后的设计目标是为树形图提供额外的空间环境，而同时保持能够探索术语分类。图6-8中，粗线条用于显示场景类型的位移，而不是刻面的位移。因此，在显示地理位移时，使用的是非常细的线条，因为在树形图中，在地理上的照片位置和叶子节点的位置之间可以描绘成千上万的线

条。图6-9给出了一个示例，在该例子中，位移向量在其节点位置终端比其地理位置终端弯曲度更大。这有助于突出任何的节点空间聚类，正如图6-9给出的在多数情况下的例子，也提供了对位移趋势的总体概览。

这种术语分类以及相对和绝对地理位置的并发视图，允许我们考虑丘陵上的小径、顶峰和石冢，同时把它们与白垩、古冢和交界点区别开。

这种空间安排可能会吸引我们注意新的关系。举个例子，马路、小路和小径表示类似的功能，但是当和丘陵一起使用时，它们分别有不同的地理特征；钓鱼和曲棍球是乡村中不同地理位置的活动；比较乡村中的礼拜堂和大教堂，或者山上的高尔夫球场和公园的高尔夫球场。图6-10显示了海滩上的一些元素的特写，这些特写帮助我们从其他方面中发现以下几点：通过潮汐的积聚场所形成的英国海岸线的的各个方面；南部地区突出了海滩内的路径和悬崖；而东南部地区突出了与海滩/元素/海港相关的特定的地理集群，以及西北中部地区突出了海滩/元素/海港。

图 6-8：对于六种选定的场景类型，其出现在地理标题和评论的描述的空间树形图。节点大小表示词频，颜色表示使用CIELab方案的绝对空间地理位置。位移向量表示非叶子节点的绝对地理位置如场景类型、刻面和描述符（见彩图18）

发现之美

我们考虑的是大规模分类数据集的空间上有序的空间树形图，该树形图的“数据/地理位置比率”很高，在审美上令人赏心悦目，而且把本章中提供的数字作为描述数据之美的备选。但是，美丽的事物应该是清晰、可用的，而且最终必须是令人满意且优雅。Matsumoto（2007）通过计算机代码环境表达了这种观点，坚信优美的代码必须具备可读性。但是，正如地图使得复杂的地理数据以适合目的的方式获得了可读性，这种方式可以决定很多空间关系，实现如航海、地理比较和模式检测的功能。因此，我们的目标是通过图形的可读性，使得Geograph项目的语言具有地理分析意义。我们能读懂Geograph数据吗？我们认为Geograph项目中的图形以及其他的图形意味着我们在诠释现有的、丰富的、美丽的地理数据集上有了不断深入发展，而且使用通过分析确定的描述和地理位置之间的各种关系获取知识，根据获取的这些知识采取措施。



图　6-9：对于六种选定的场景类型，其出现在地理标题和评论的词汇的空间树形图。位移向量表示非叶子节点的绝对地理位置（共现词），并提供了关于位移的空间聚类和空间趋势的信息来满足树形图的空间填充目标（见彩图19）

这些场景视图中描述的空间关系依赖于宽高比和叶子节点的平均地理位置。我们可能会对比这里所描述的更精确的地理位置感兴趣，而且数据密集的树形图对于为更传统的映射选择候选的刻面和描述符一直都很有用。通过树形图推导出的词汇共现图能够帮助我们确定在术语使用上可量化的空间区别。举个例子，我们发现在南部和西南部地区，山谷和丘陵共现的频率比期望的还高，而在北部地区，顶峰和丘陵一起使用更频繁。而且当使用Geograph项目的数据中其他可选的描述方式时，这

些关系在不同程度上一直存在。它们可能会反映在选择需要记录的场景的特定方面有偏好：区域地理特征、语言区别、在特定地方贡献图片的个人偏好，或者以上这些特征的结合。不论怎么理解，我们对树形图的理解，由于其他地方的启发，引起了我们对一些其他地方的地理特征的兴趣。同时需要注意的是这里使用了一些较生硬的词汇来描绘乡村。这些趋势可能并非是有意识的偏见，而可能是和为Geograph项目选择一些令人赏心悦目的景观方面相关——这意味着我们对这些数据的理解使我们更加坚信处理的数据之美。我们打算继续探索和分析这种关于地图和更正规的空间地理术语。用户选定的描述性术语的地理特征的以上及其他方面，有助于描述地方的“本体”特征的发展。

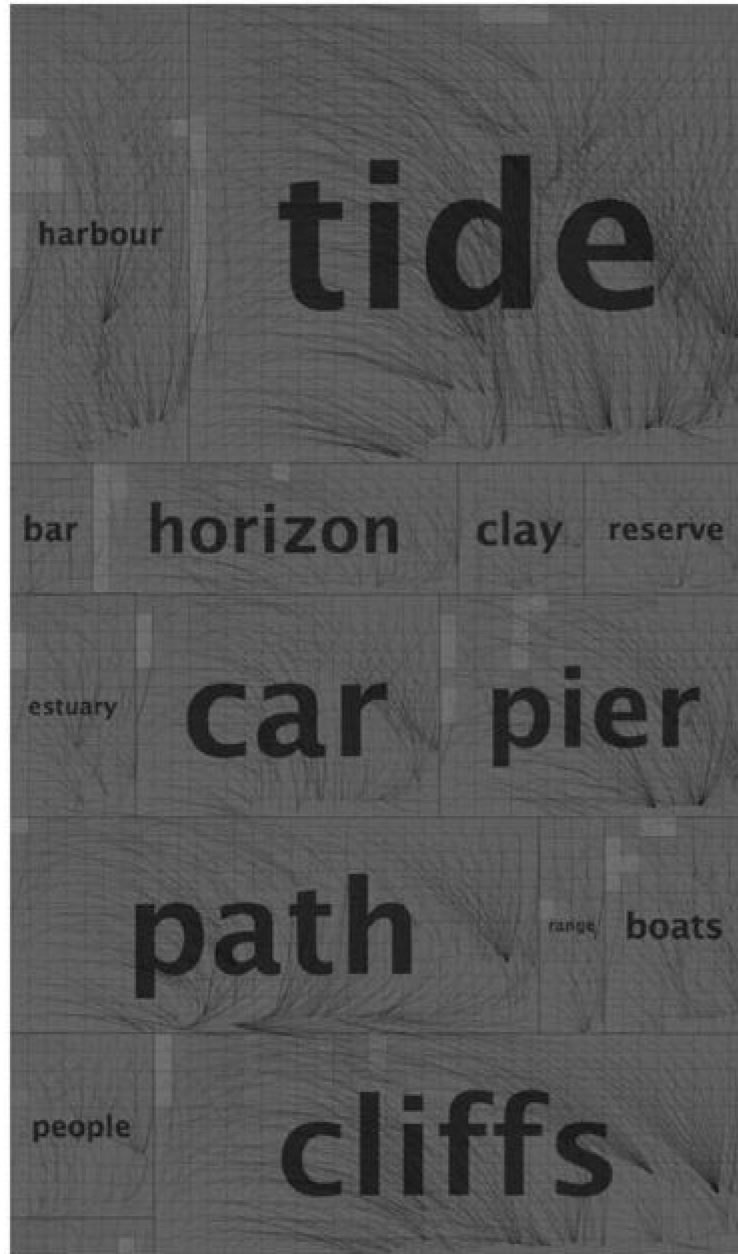


图　6-10：在beach基础区块中出现的对选定元素的描述符在地理标题和评论中出现的术语的空间树形图。位置向量表示图6-9中放大部分的叶子节点的绝对地理位置（见彩图20）

我们的图形和探索是不全面的。我们正在研究由社区参与者贡献的地理信息中存在的系统偏见所造成的影响，并提出策略来减轻这些影

响。我们正在制定标识符，用来描述视觉设计空间和交互式应用，通过这些标识符可以做出进一步的探索。我们尚未考虑在Geograph项目中所确定的在地理上不同的关系是否会随着时间推移而依然保持一致。然而，这里所描述的方法帮助我们获取知识，克服当考虑规模、结构、多样性和复杂性时会面临的各种不利条件。但是，除了居住在某个地方的大量的志愿者共同努力外，谁还能更好地帮助我们了解这个地方？此外，除了通过精心设计的多元图形来揭示结构和帮助发现，如何能更好地探索在描述范围之内的某个地方的意义？

反思和结论

我们认为数据之美在于其深度。当先前隐藏着的结构和模式开始展现出来时，美丽就显现出来。这些模式激发了人们对数据提出新的想法和质疑。数据激发人们、鼓励探索并为探索者提供洞察力。只要存在空间数据，就可能呈现地理之美。

美丽的数据激励人们实现优雅的数据可视化策略，正如它也激励探索并回报那些探索者。可视化特别适合于探索地理模式，正如好几个世纪的制图学所证明的。

我们这里给出的例子说明了我们可以如何使用美丽的数据，如Geograph项目的数据的地理特征，从以一个视图的方式到使用为信息管理和检索的应用场景的描述来描述一个地方。复杂、数据密集且优雅的图形是该过程的重要部分。

我们广泛地认为美丽是一个实体的特征，它给人们提供快乐、意义或满足。对于数据及其表示，Geograph项目的各个方面都包含了这些特征品质。我们对包含高度的“数据/地理位置”比率，通过空间填充图形的Geograph项目的某些方面所做的可视化，是富于创新、有创意的，而且具备解决问题的基础。它有助于沟通同事间的工作，并开辟新的分析途径。我们将及时发布人们对大不列颠群岛的人文和自然地理这些数据密集的描述，作为我们所创造的最美丽的数据。

致谢

我们对Barry Hunter和Geograph British Isles(http：//www.geograph.org.uk/credits/2008-04-31）的贡献参与者表示衷心感谢，他们的工作基于Creative Commons Attribution-ShareAlike 2.5 License(http：//creativecommons.org/licenses/by-sa/2.5/）可以获取。

我们对Ross Purves和Ross Purves在Geograph项目中的工作付出也深表谢意。该项目是TRIPOD项目的一部分，由欧盟委员会支持，合同号是045335。

参考文献

[1]AndrienkO'G.，N. AndrienkO'J.Dykes,S.Fabrikant,and M.Wachowicz.2008."Geovisualization of dynamics,movement and change:key issues and developing approaches in visualization research."Information Visualization,v.7：173-180.

[2]Bertin,J. 1983.Semiology of graphics(W.J.Berg,trans.）.Madison:University of Wisconsin Press.(Oiginal work published 1973.）

[3]Cawthon,N.，and A. vande Moere.2007."The Effect of Aesthetic on the Usability of Data Visualization，"Proceedings of the 11th International Conference Information Visualization,IEEE Computer Society Washington,DC：637-648.

[4]Edwardes,A.，and R. Purves.2007."A theoretical grounding for semantic descriptions of place."M.Ware and G.Taylor(eds.）.LNCS:Proceedings of 7th Intl,Workshop on Web and Wireless GIS,W2GIS：106-120.

[5]Fabrikant,S.，M. RuoccO'R.Middleton,D.MontellO'and C.Jörgensen.2002."The first law of cognitive geography:Distance and similarity in semantic space."Proceedings of GIScience 2002：31-33.

[6]Fisher,P. F.1998."Is GIS hidebound by the legacy of cartography?"The

Cartographic Journal,v.35：5-9.

[7]Geograph."Geograph British Isles-photograph every grid square."http：//www. geograph.co.uk/(acessed April 9，2009）.

[8]Goodchild,M. 2007."Citizens as sensors:the world of volunteered geography."GeoJournal,v.69：211-221.

[9]Hawgood,D.，D. Dunford,R.Farrow,B.Hunter,and P.Mayes."Geograph or supplemental."http：//www.geograph.org.uk/article/Geograph-or-supplemental/(acessed April 9，2009）.

[10]Kernighan,B. W.2007."A Regular Expression Matcher，"in Beautiful Code:Leading Programmers Explain How They Think,ed.Andy Oram and Greg Wilson，1-9.Sebastopol,CA:O'Reilly.

[11]Kosara,R. 2007"Visualization Criticism-The Missing Link Between Information Visualization and Art."Proceedings of the 11th International Conference Information Visualization,IEEE Computer Society Washington,DC：631-636.

[12]MacEachren,A. M.，and M.J.Kraak.2001."Research challenges in geovisualization."Cartography and Geographic Information Science,v.28：3-12.

[13]MatsumotO'Y. 2007."Treating Code as an Essay，"in Beautiful Code:Leading Programmers Explain How They Think,ed.Andy Oram and

Greg Wilson，477-481.Sebastopol,CA:O'Reilly.

[14]Oram,A. and G.Wilson,eds.2007.Beautiful Code:Leading Programmers Explain How They Think.Sebastopol,CA:O'Reilly.

[15]Shneiderman,B. 1992."Tree visualization with tree-maps：2-d space-filling approach."ACM Transactions on Graphics(TOG)，v.11：92-99.

[16]Skupin,A. and S.Fabrikant.2003."Spatialization Methods:A Cartographic Research Agenda for Non-geographic Information Visualization."Cartography and Geographic Information Science,v.30：99-119.

[17]Tufte,E. R.1983.The Visual Display of Quantitative Information(First Edition).Cheshire,CT:Graphics Press.

[18]Wood,J. and J.Dykes.2008."Spatially Ordered Treemaps."IEEE Transactions on Visualization and Computer Graphics,v.14：1348-1355.

第7章　数据发现数据

Jeff Jonas和Lisa Sokol

简介

新一代的“智能”信息管理系统将不会依赖于用户向计算机提出一些异想天开的问题；相反地，它们会自动确定新观察到的现象是否可以表示有足够理由去做出某种响应，如给用户或系统发送关于潜在隐患或者危险的自动通知。

一个组织最多只会和它的所有成员的洞察力总和一样有智慧。这些智慧源于观察——通过各种企业系统收集到的观察，如客户登记系统、财务会计系统和薪水系统。每一次交易，企业组织都可以从中学到一些东西。因为在某个时刻获知了一些信息，才有了一些机遇来弄清该新数据的涵义并做出适当的响应，实际上这也是一种义务。举个例子，客户记录的当前地址变化是否表示该客户和你的最重要的50个客户之一相关？如果一个企业组织不能评估新的数据点如何实时地与其历史数据相关，那么该组织将会错失采取行动的机会。

当“数据可以发现数据”时，存在着通过洞察力找到用户的机遇。

数据如何发现数据是关于数据可发现性的声明，是对已有的信息和新的数据进行定位和关联所能达到的程度。可发现性要求能够召回相关的历史数据，这样新到达的数据可以定位到该位置，这和每个拼图游戏的方式相似，它们是通过其他相关的正在进行中的谜题来评估的。每个新的谜题都是增量式地构建在已知的方面，每个给定点在时间上和不断发展的拼图相关。通常来说，新的信息虽然对于构建全局大图很重要，

但是它们自己本身并没有带来新的至关重要的信息。（相反地，有些信息可能会改变谜题结构，这种改变导致产生报警——结果发现是由于某个拼图放置错位）。正是在这个时刻，当新的谜题能够重新改变图像的结构，发现就产生了。实时发现不需要用户三思而后行并在恰当的时候提出正确的问题。

那些无法切换到“数据发现数据”模式的组织，其竞争力更差且效率将会更低。

实时发现的好处

立足于“数据发现数据”的高级信息管理系统，不会依赖于用户向计算机凭空提出正确的、相关的和即时的问题。虽然该技术会引发新的策略探讨，如允许哪些数据去发现哪些数据以及什么级别的相关度应该通知谁，以下是“数据发现数据”系统可以做什么的一些例子：

客户便利

在酒店辗转反侧一宿后，客人在早上7点时终于决定打电话要求晚点退房，并预约了中午的电话叫醒服务功能。客人刚沉睡不久，服务员无意敲门准备打扫房间，于是发生了令人不愉快的事情。遗憾的是，全世界的酒店旅客，这种骚扰经常发生。当数据发现数据，延迟退房和叫醒电话需要和服务员调度信息保持一致。这种“数据发现数据”的事件会触发一条自动的文本信息，通知服务员在下午2点前不要打扫该房间。

客户服务

某个用户对于一本即将出版的新书感兴趣，他在Amazon搜索该书标题，但是没有找到。该用户决定每个月都检查一下，直到该书出版。遗憾的是，该用户下一次检查时，发现该书不但已售空，而且需要重新预订，等待第二次印刷。如果能够做到数据发现数据，一旦该书开始销售，该数据检查点就会发现该用户的最初请求，自动给用户发送邮件，告诉他该书已经开始销售了。

加强儿童安全

有位妈妈想要确保她的小孩在步行去学校的路上是安全的。这位妈妈可能会搜索该社区的Web站点，确保没有登记的犯罪分子居住在孩子去上学的路上经过的小区。为了确定在孩子上学途中的街道上是否增加了一条新的犯罪分子地址，妈妈会每天都去上网检查一下Web站点吗？如果使用“数据发现数据”模式，在孩子上学路上一旦有新犯罪分子在某个小区登记住址，这种新的数据会自动链接到先前的查询。这位妈妈就可以立即知晓相关的发现。

越过“划分”，进行探索

有些部门使用“划分”，故意把数据隔离起来。隔离数据有助于防止高度敏感的数据意外泄露出去。尽管新的政策授权对信息共享，但是对高度分类的数据的传统保护措施，阻碍了部门发现两个“划分”正在处理相同的主题或者在主题上有交叉。其中一个例子可能是某个机关正在执行反恐，而另一个正在做反毒工作。部门有数百个“划分”，而一个人要在另一个“划分”定位相关数据是很不现实的，因为他永远都不知道其他人会有什么信息。如果数据发现数据，当有一条记录加入到反毒部门的数据库中时，如果该记录和反恐部门相关（如涉及同一个人的数据），系统就会立即发送通知给相关人士。

赌桌上的舞弊

这是一个真实的故事，坏人发现了好人的数据——导致一个意外的发现并产生令人惊喜的结果。

在20世纪90年代中期，美国很多新的行政管辖区承认了河船赌博(rverboat gaming)是合法的，路易斯安那州就是其中之一。一个新的游戏管辖区所面临的挑战之一是缺少一批有丰富游戏经验的当地雇员候选人。因此，新的管辖区必须培养一批有广泛的专业工作类别的当地劳工，岗位包括从交易员到监察室业务员。路易斯安那州的Bossier市就是这样的一个社区，它需要从无赌场业务过渡到赌场河船业务。

今天和任何其他赌场的任何其他日子没有什么区别。交易员正在监视着赌场的玩家。该楼层的主管正在监视着交易员和玩家。赌场经理正在监视着楼层主管、交易员和玩家。而监察室正在监视着每个人——甚至是该赌场的执行官。监察室有责任监视游戏的交易情况，不仅仅是为了保护赌场，而且也是为了保护客户。监察室同时聚焦于游戏的交易以及捕获其他犯罪行为的证据。例如，抢钱包的凶徒属于违法行为，因为他干扰了用户体验。

监察是最后一道防线。

在一间非常小的监察室里，安装着数百部摄像机（在拉斯维加斯的大型赌场有数千部）。一面墙上可能安装着20、30甚至40部的监视器，看起来像是在做犯罪现场调查(Cime Scene Investigation,CSI)。因此，就

这么几个操作员审查进入房间的权限，怎么可能有意义呢？答案就是：绊网（暗线）和周密的浏览。

绊网(Tipwire)有很多种形式，包括从热线上的一条建议到一个楼层的主管要求监察室评价一个客户的打牌行为，例如，确定她是否在对桌牌计数[1]。这一天，监察业务员在随意查看——即执行所谓的随机审查——从一个表到另一个表，简单看一下，检查是否有不正常的，然后继续查看下一个表。

等一等！那是什么？简直不可能！悉心观察的监察业务员刚刚观察到一个玩家公然在轮盘赌桌(rulette table)上作弊。今天所发现的这个骗局称为“逾期投注”(pst posting)——某个玩家在轮盘已经停止滚动，停留到某个数字后才下赌注。“逾期投注”发生在当玩家注意到滚球已经停止到某个数字（如32），发现这个结果后，马上押注一个逾期的、必赢的赌注。“逾期投注”和桌牌计数不同，后者是合法的但是不鼓励，而前者实际上是舞弊，而且绝对是不合法的。

今天很特别的原因是那张桌子只有一个交易员和一个玩家。通常，“逾期投注”舞弊方式一般会涉及很多玩家——这些玩家一起努力阻止交易员监察到逾期投注的发生。这种团体活动可能会涉及同一张桌子的两个或者更多的“玩家”，他们看起来似乎完全不相关（比如，表现得彼此都不认识对方）。等到骰子滚到某个数字不动了，一个玩家（如一个看起来性情很好的妇人）会沿着整个桌子假装要下赌注。交易员当然会说：“女士，您下的赌注太晚了。”但是，她的这个下注对必输的号已

经使该桌的其他成员（如一个看起来像摇滚队的、留着莫霍克发型的花花公子）对必赢的号下了一个迟到的赌注——而且避免了被交易员所发现。

安保人员得到通知，警卫小组也调动起来。该玩家遭受逮捕和拘留。怎么会发生这样的事呢？交易员显然很尴尬。在新的行政管辖中，这名雇员还是一名游戏新手，而且确实只受到过课堂培训。交易员清楚地表示，对于这种类型的诈骗，她只是听说过而已。然后，她说安全报警今天确实做得很不错。她非常尴尬，而且很快保证，在她以后的监察期间，再也不会发生这样的事了。

输入“数据发现数据”。

当骗子被逮捕时，他必须证明自己的身份。今天的骗子和任意某天没有什么区别，介绍了他的名字、地址、电话号码，以及一些其他信息。这种信息收集是通过一个标准的表单，包含一个签名（如果他们能得到），然后，数据进入企业安全逮捕处理系统。

当然，这个骗子和交易员的姓氏不同，否则事情就会变得非常简单。当然，该骗子的地址和这位交易员的地址也不一样。但是，他的电话号码恰巧就是该交易员求职应聘时用的电话号码。正在这个时刻，逮捕信息发送给逮捕处理系统，该系统是实现数据发现数据的次级系统[2]，它找到这个至关重要的发现，并生成一条即时报警信息：“5764号员工和44-00321号被逮捕人有相同的电话号码！”

长话短说，交易员受到对质、开始招供，于是该交易员和骗子一起

被交给执法机关接受起诉。

需要明确说明的是，用户并没有收集之前作弊诈骗者的身份属性（如名字、地址和电话号码），并尝试搜索大范围的业务系统（称为联合搜索）。求职者、员工、会员俱乐部和逮捕处理系统等，这些系统甚至不能通过地址或电话号码来搜索——它们不是为搜索而设计的。

在“数据发现数据”的环境中，用户不需要主动搜索或者提出相关的问题。安全部门的用户在系统中输入他们学会的东西，然后通过其他企业信息资产来评估这种新的信息。发现新的信息需要和已有的数据相关，而且这种关系满足预定义的兴趣条件：有个“坏人”和某个员工相关。因为坏人和该员工共享一个电话号码（所以这个坏人认识这个员工），这样就满足该兴趣条件，并触发一条报警信息。如果发现这个骗子逮捕相关的数据和三年前某个宾馆的一个顾客相关，这种不满足兴趣条件的发现不会生成报警。

为了说清楚这个观点，我们现在假定这个“逾期投注”者提供的电话号码和这位交易员的电话号码没有任何关系。在这种情况下，就不会生成任何报警信息。交易员可能会受到某些质疑，但是显然没有足够的理由来起诉她。可能公司的安全部门会调查这个交易员，或者雇佣一个私人侦探来调查这两个人是否是朋友？谁知道呢？

但是，如果他们真的有关系呢？要是电话号码不匹配，系统认为他们没有关系，那么发生什么事呢？交易员继续从事交易，时间继续推移。如果六个月以后，交易员在她的人事系统中更改了她的家庭住址，

而且她的新信息和那个骗子的地址完全相符，企业组织如何能够知道这件事呢？实际上，没有一个企业组织会发现这一点，除非它能够做到“数据发现数据”。一旦新的信息能够和之前已经结束的案件关联起来，该系统就可以检测到报警条件：坏人和某个员工相关。

顺便说一下，报警并不意味着真的有犯罪活动。但报警确实为企业组织集中有限的调查资源发挥了重大作用——在这种情况下，基于足够的信息条件，给更相关的可能事件报警。

其他例子是：在这个轮渡赌场行业，还发现和一个市场营销人员相关的骗局，这个营销人员提出了一个系统攻击、返还现金比赛的想法，（在营销活动中，你玩得越多，就会获得越多的积分，而且这些积分可以兑换现金），而结果是那些现金都是返回给他的室友。而在另一个案例中，某个人举办“一天一辆车”活动抽奖，系统发现中奖的人“碰巧”是他的妹妹（只是姓氏不同）抽中了该车，这两个家庭成员在现场时演得好像彼此从来都不认识对方。

通过“数据发现数据”的策略，以上三个骗局最后都真相暴露——所有这三个骗局都是在90天之内就被查出个水落石出！

数据发现数据非常重要的原因是因为信息到达的时间顺序是不确定的。想当然地认为事件是有序的系统和处理方案有一个致命的缺陷：无序的事件可能会提供企业组织足够的信息，但从来都没有基于这些信息真正采取措施。这一点在后面会谈及更多。

一个在美国有几千个连锁店的大零售商分析了它的历史数据，很震

惊地发现它的每一千名雇员当中，有两名实际上曾经在该店偷窃过，并被逮捕。更糟糕的是，这些员工所被抓到的店和雇佣他的店是同一家店。虽然数据表现方式不同，一旦企业有了这些证据，就一分钟都不能耽搁。该连锁店需要立刻意识到这一点。

解决无序的数据点的传统解决方法十分繁琐。企业安全部门应该如何利用新的数据，使这些数据能够揭露一个员工可能曾经就是偷窃者？何时在人力资源系统中更新一条员工记录会导致企业安全部门重新评估它之前所有的调查呢？这种重新评估该如何结构化，这样当在内部调查数据库中，有新的或者更新的记录和员工相关时，企业才不会错过这些信息？一种策略是周期性地测试调查的数据库和整个员工数据库。另一种策略是企业安全部门定期重新调查每个员工。但是这两种策略都会错过一些重要的发现，因为定时（时间）是至关重要的。

即使是完美的算法，当它在基于完全重新设计的业务系统上运行时（如人力资源系统和内部调查系统），它依然会错过一些可发现的事件。如果为了确定两个人是同一个人或者相关所需要的数据存在于一个完全不相关的系统中，那么又会发生什么事呢？举个例子，想象从一个不相关的系统中得到了第三条记录，比如会员俱乐部登记系统，它反映了家庭地址和家庭电话号码之间先前未知的链接。什么样的企业系统会用于检测这种被我们称之为“非显著关系”(nnobvious relationship)的条件？

数据发现数据，包括非显著关系条件，要求人们首先解决“企业的可

发现性”问题。

[1]值得注意的是，赌场监督的作用是提供情报。他们报告自己的观察和发现，但是没有执行力；执行力是安全部门所发挥的作用。

[2]这种被称为“无明显的关系意识”(Nn-Obvious Relationship Awareness,NORA)技术，由Systems Research＆Development，即SRD公司（由Jeff Jonas成立）为拉斯维加斯的游戏行业开发。SRD公司从那以后被IBM收购，现在是IBM的实体分析部门的一部分。这里有一些额外的信息——IEEE论文：Threat＆Fraud Intelligence-Las Vegas Style(http：//jeffjonas.typepad.com/jeff_jonas/2006/11/ieee_paper_thre.html)

企业的可发现性

当企业中有新的信息到达时，不论是来了一个新员工、员工的某条记录更新、剽窃者信息或者是会员俱乐部登记，人们需要记住其他什么组织数据和这个信息相关。

企业发现的一个常见办法是采用“联合搜索”技术，把查询传递给每个相关的业务系统。正如我们所证明的，这种方式对于数据发现数据系统并不适用。可发现性，尤其是在大规模、实时的环境中，目录是必需的。

联合搜索不是万能的

企业组织有众多的业务系统，每个系统都有自己专用的业务功能、定制的信息结构、分析和报告。二级数据聚合是很常见的，而且包含如数据仓库、业务数据存储和数据集市。数据聚合存在无数的信息孤岛，每个孤岛都和特定的任务或功能相关。

传统的联合搜索系统涉及用户查询每个孤岛的数据库，获取相关内容。更复杂的联合搜索系统使用智能中间件，为数据库的每个查询做代理；智能中间件是一个模型，在该模型中，中间件通过自动结合各种信息来处理查询，并对查询的结果进行编制，返回集成的结果集给查询者。

在需要信息的时候，联合搜索“即时”收集跨孤岛的数据。虽然这种类型的联合搜索在一些场景下是可用的，但是无法很好地适合于高性能

的企业可发现性，后者需要根据数据发现数据来生成。

有两种主要的原因使得联合搜索无法扩展：

现有的系统通常不包含高效定位一条记录所需的必要的索引。如支付系统通常会包·含预先构建的索引（对数据定义指针），从而有助于在员工号码、纳税ID号码和名字进行查询。支付系统很少会包含高效的方式来定位地址或者电话号码上的记录。假设新发现的赌博诈骗者暴露了他的地址和电话号码。我们的支付系统保留所有员工的电话号码和地址记录（毕竟它需要发送支票）。但是该支付系统无法简单地生成（如果它可以）员工的列表，这些员工和之前提到的赌博诈骗有相同的地址或电话号码。即使在支付系统中为员工的电话号码中创建了一条索引，但是它仍然无法使人们在相同的系统中定位紧急联系电话号码。如果我们无法对赌博诈骗者的验证信息和所有相关的员工数据进行比较，我们还是错过了发现员工和诈骗者之间的联系。

如果某个字段不在索引中，在数据库中定位一条记录的方法称为“表扫描”。在表扫描过程中，对搜索的值和表中每条记录的值进行比较——从数据库中的第一条记录开始比较，到第二条，等等。因此，数据库越大，每次搜索花费的时间越长，因而主服务器系统上的计算压力就越大。

更糟糕的是，联合搜索需要递归处理（某些情况下需要重复一些步骤），这对于分·布式查询环境简直是一场噩梦。假设你执行一条联合查询，是为了发现和某个特定的人相关的企业记录——也就是说，根据

某个特定的人的名字和出生日期开始查询。如果联合查询返回这个人的一些新的属性，如一些地址和电话号码，那么你就了解了一些新的东西。为了更全面地了解，需要最大化利用所了解到的关于这个人的新数据，也就是说，初始化另一个企业范围的联合查询，防止基于这些新的数据点，定位额外的数据。因此，如果第二次的联合查询发现了另一个地址，拼写名字的多种方式，以及一两个别名，那么又会发生什么情况呢？更具体全面地说，每次了解一些能够发现先前丢失的记录的知识，发现过程必须执行另一个企业范围的联合查询。

下面这个真实的案例强调了这一点。某个企业组织（一个商业实体，不是政府）有2000多个数据库。用户查询被定向到这些数据库中来收集相关的记录。为优化搜索过程而设计的优秀的中间件产品需要花很多年的时间，以及几百万美元的成本。这种智能的系统会识别哪个数据库应该处理哪个查询，确定合理的数据库访问方式，把查询同时传播给所有相关的集合，并汇集了解到的知识。但是，这种联合搜索方式永远都无法克服一个严重的设计缺陷：每次发现了解一些知识（如一个别名或者一个新的电话号码），优秀的中间件必须重新把这些查询发给很多数据库。这种递归过程运行在大规模的计算机集群中，最终需要定制为每8分钟停止处理。注意下一次的递归查询可能最终会发现一条重要记录。

但是，需要8分钟！这意味着用户或者系统只能等着，无法执行任何操作，因为在这段期间内无法获取到任何答案。然而，在数据发现数据

的情况下，数据是个问题。这意味着每当有新的数据到达，在处理这些数据之前会有最多高达8分钟的延迟。想象一下，在整个企业网络，每秒提交成百上千的联合查询的规模下，通过无数的业务系统递归执行，使用联合查询这种方式是多么得不切实际！

如果以上提到的因素还不具有足够的吸引力，联合搜索的致命一击是所有必须搜索的系统在物理上必须是可访问的，而不是处于维护或者备份状态，或者处在周期性的半夜或者月末的批处理过程中。当然，连接性也必须是完全可运作的。考虑这些需求以及由成百上千的涉及建筑、时区和大陆的系统组成的组织。联合搜索无法支持“数据发现数据”的任务，因为它无法大规模地基于企业的发现力来发布数据。

目录：无价之宝

想想一个图书馆。把图书馆的地板、走廊和书架想象成存储信息的孤岛。有价值的信息被搁置一边，等待着我们去发现。人们不需要为了寻找特定的一本书而走遍图书馆大厅。相反地，人们会使用卡片分类目录，在主题、标题和作者上执行交叉索引来帮助查找相关的文档。目录、索引和分类目录基本上指的是相同的东西：都是用于定位其他东西。

定位器的例子包括图书馆的卡片目录、电话号码目录和eBay等。对于每种定位器，目录相当于定位器服务：当对它们提供一个或者多个查询项时，这些定位器会返回引用信息（指针）。在图书馆，卡片分类目录是为了促进高效的企业搜索的一个特定目录，它为用户提供指向文档的特定的指针（如：使用Dewey十进制系统）。当提供给用户一个指针后，就变成“联合搜索”。注意（有用的）联合搜索和（无用的）联合搜索之间的区别。

G公司的搜索也不是扫描整个地球来获取搜索结果；相反，它是搜索G公司创建的一个特定目录，然后把搜索结果指向真正文档的指针返回给查询者。

企业范围的可发现性的唯一可扩展解决方案是使用目录。这并不奇怪。因此，一个特定的目录是最根本的部分，它允许“数据发现数据”系统有规模地处理可发现性和实时地确定相关性。

不是所有的目录都是等同的。传统的“上下文无关”的目录和能够累计并持久的上下文的目录之间存在很大的区别。上下文相关的目录使得数据发现数据可以通过非常令人意外（不显著）的、实时、大量的方式，而且效率非常高效。

上下文无关的目录是最常见的目录类型：每个文档的索引和所有其他文档不相关。换句话说，新的企业交易（文档）更新目录，并不关心该交易（索引的元数据）和其他交易之间的关系。上下文无关的目录是为了给用户提供最基本的定位文档的能力而设计的（如所有和“Billy the Kid”相关的书）。

语义上一致[1]的目录是那些试着利用同义词的目录，使用不同的单词来表示相同的东西。这意味着用户想要寻找一个东西（如“Billy the Kid”）应该能够找到其他“相同”的东西（如“William Antrim”，他的一个别名）。当一个新报告的实体引用了之前观察到的实体，语义一致的目录能够做出识别。包含语义一致的数据的目录可以看成是图书馆的一个卡片文件，这有一个很大区别：和类似的实体相关的卡片被胶合在一起。这意味着如果一个搜索定位到了一张卡片，“作为奖励”，它不需要付出额外的努力就可以发现所有其他相关的卡片。最显著的是，在这些胶合在一起的图书馆卡片中，有些甚至不包含被搜索时的原始数据项。

坦白地说，这一切看起来很神奇。当试着去发现该企业对于某个Email地址的了解程度时，查询者会发现和该Email地址相关的记录以及关于这个人在企业中的其他记录——举个例子，某个会员俱乐部，虽然

实际上它的记录中从来都没有包含Email地址。语义上协调身份的算法（如人们或者组织）有时被认为是身份识别[2]。确定多个实体什么时候是相同的实体的算法需要非常深入细致的讨论，因而超出了本章讨论的范围。

语义一致且“有关系意识”(rlationship-aware)的目录是提供了更高程度的上下文的一种目录类型，它通过允许用户发现额外文档，如那些和私人相关的关联（如Billy theKid就是William Antrim)。可能知道实际上确实有William Antrim可能也是很重要的，而且这个人碰巧是Billy the Kid的继父。对这些达到可视化的方式是在脑海里画一张图书馆卡片目录，其中一些卡片如前所述，已经通过胶带系在一起，还有一些线条连接到一些卡片捆绑(bndle)到其他卡片以及其他卡片捆绑。人们可以搜索任意术语的某个集合，并定位一个卡片捆绑，并且在那时弄清楚该捆绑是如何和其他捆绑关联（相关）的。有些关联的线条比其他线条粗，这表明它们关联关系更强。通过一条线找到另一个卡片捆绑，人们可以马上看到是什么线条连接到下一个捆绑。通过这种方式，人们可以观察到这些卡片捆绑分离的程度，如Kevin Bacon所提出的六度分割理论。

当思考“数据发现数据”系统时，必须记住当有新的交易发生时，可能是把当前已经存在的实体增加到上下文中，那么就有可能会发生由于捆绑和线条的重新组织，新的信息改变了图片的大小。

持久性上下文(Prsistent context)是个术语，用于表示一致的、“有关系意识”的当前状态。从本质上讲，它指的是不断变化和越来越复杂的

当前状态。持久性上下文是保存了事物间相关性，优于在联合搜索系统中用到的“即时”性上下文(Jst-in-time context)。

持久性上下文（语义上一致且“有关系意识”的路径）能够促使高性能的发现、流语境化以及实时检测相关性。同样值得注意的是，如果相关性检测可以在获取数据时发生，其计算代价是最低的。从这方面来看，图书管理员（负责把新数据补到目录中）是第一个发现新到来的数据和待发布的数据是否有足够的相关性。

意识的溪流胜过沸腾的海洋。想象这一点的一种清晰的方式是想象一个企业组织有4 exabytes（1 exabytes=2^{16}字节）的历史数据，每分钟接收5TB的数据。你觉得他们是在周末启动一个进程，用来发现发生的事吗？在整个地球上，可能没有足够的计算机或者能源来做这件事。（注意：规模上设想的行为在设计非常高效的系统时，被证明是非常有效的。）

[1]定义为“当两个东西描述上虽然不同，但实质相同，能够做出识别”。

[2]不要把身份识别和类似的有时被称为匹配/归并或归并/清洗的一组方法混淆。更多关于它们之间的区别，可参考文章“Entity Resolution Systems vs.Match Merge/Merge Purge/List De-duplication Systems”(http：//jeffjonas.typepad.com/jeff_jonas/2007/09/entity-resoluti.html)。

相关性：什么是重要的以及对谁重要

现在，有大量的数据涌向我们。花点儿时间来评估在你周围环境的所有视野和声音：背景是计算机的嗡嗡声，音乐的播放，电视上的漫画以及孩子们想要偷窃彼此的玩具。人类本身有自己内部的相关性检测算法，可以评估所有展现的数据，并且当某些观察使我们注意并且可能做出某种反应时，会提醒我们。就如忽略小孩们以及他们的吵架，直到该吵架上升到暴力威胁。正是某个小孩的一个动作，提醒我们是时候开始出面调解。

人们如何使用技术来计算相关性，以及什么时候允许它自动注册一个报警是至关重要的。在技术方面，目标是一种状态，在该状态中，报警队列中的下一项是审查时最重要的下一个项目。没有理由产生的报警比现有的可用于应付它们的资源还多。例如，风险评估引擎生成的报警数必须被配置为和某个人的个人风险、编制和反应能力相适应。如果资源增加了，人们可以提高报警的灵敏度。

各个组件及特殊考虑

智能的“数据发现数据”的环境必须包含8个最基础的组成部分：

·观察的存在和可用性；

·能够从观察中提取特征并对特征进行分类；

·能够有效地发现相关的历史背景；

·能够对新的观察做出判断（相同或相关）；

·当新的观察推翻先前的判断时，能够做出识别；

·能够积累和坚持主张；

·能够识别相关性/洞察力如何形成；

·能够通知相应实体的洞察力。

观察的存在和可用性

如果没有数据，人们就不可能会理解观察；而如果有数据，为了使这些数据某天有潜在的利用价值，那它就必须通过一些传感系统来“感应”（收集）。而且即使数据被收集，人们为了还有一线希望能够理解它，必须能够访问它。

能够从观察中提取特征并对特征进行分类

为了避免不必要的争论，我们说一粒沙子包含的用于抽取和分类的特征太少。沙粒往往具有相同的颜色、大小、重量、形状等。因此，缺少可区别的特征后期将会妨碍人们识别相同的数据（语义一致）。其要点是，为了把数据放到语境中，人们必须能够抽取和分类它的关键特

征。当地址信息包含在一个列中，而姓名在另一个列中时，结构化数据是非常容易的。无结构化的数据，如新闻报纸和博客，花费的时间更多；提取正确的名字和地址是非常具有挑战性的，它通常被称为实体提取。从视频中提取特征，如为汽车拍照的读者，可以在某些情况下进行。

长话短说，人们如果想要把观察结合到上下文中，必须能够对观察的主要特征进行提取和特征分类。

能够有效地发现相关的历史背景

随着新的观察到达，抽取和分类的关键特征被用于查询上下文相关的历史数据（我们称之为持久性上下文），用来发现这些数据是否适合。为了能够实时发生，支持大体积的数据流，这种发现需求必须是极度快速的。

能够对新的观察做出判断（相同或相关）

当有新的观察根据历史数据语境化(cntextualized)时，该算法必须能够做出以下断言之一：1）新的实体，是其类型的首例（如一个新人）；2）已知的实体（如是我们已知的关于某人的观察），在这种情况下新的实体通过已有实体“分解”；3）该实体（新的实体或者是已知的实体）如何和其他实体相关。

因此，存在这样的观点，正如人们把拼图玩具拼在一起，其结论是没有什么可以做的。正是这时，人们抛弃了当前项，不管它曾经在什么位置，而且把注意力转移到下一个拼图模块（观察）。注意：可能你的

最后一次判断是错误的，但是关于这一点，你不太可能会发现它，除非有新的拼图模块暴露出了这个潜在的错误。

当新的观察推翻先前的判断时，能够做出识别

有时一个观察包含新的信息，证明之前的判断必须完全推翻。可能这种新的信息证明之前认为不相同的实体实际上是相同的实体。相反地，一个新的观察可能会暴露人们之前认为相同的实体，而现在坚信它们一点都不相同（例如，新的数据点说明了我们数据库中的两个Bob Joneses不是同一个人，而实际上一个是大三的，一个是大四的）。

使用新的观察推翻先前的判断是语义一致算法最复杂的方面之一。但是如果没有这个关键特征，数据库会随着时间流逝，和真实情况越来越不相符。它的缺点是使用周期性的数据库重载来纠正这种问题。而且对于非常大的数据集，很显然这会带来可扩展性噩梦。

能够积累和坚持主张

当完成断言过程——换句话说，当新的观察被规约为断言（新的、相同的或者相关的实体），而且新的观察是用于弥补和之前相同的断言——新学到的知识必须放到数据库中，这样下一次交易可以从新的知识中受益。在某些方面，这感觉上很像最基本的增量学习系统。

能够识别相关性/洞察力如何形成

只有当新的观察应用于历史数据，这样无法保证更多的计算，该系统该“扪心自问”，“我是否学会了一些真正有用的东西？”，它和一个人不断地查看在每次放上一个拼图模块时会有什么发现很相似。

我们需要做的工作包含检测预定义的兴趣模式。例如，发现一个好人是否认识一个坏人是相关的，或者某个人每天的现金交易是否超过1万美元。

但是，新的相关性参数可以基于外部过程来设置，它可能包含人类洞察力或者次要的模式发现/数据挖掘引擎。

能够通知相应实体的洞察力

当检测到某方面的洞察，应该通知谁或者哪个系统？在我们现有的实现中，这是小事，因为每个相关性规则（比如，如果一个可能的主顾是我们的最高的50个客户的一位很亲密的同事），以及传播规律（也就是说，发送一条友好信息给赌场主人）是同时建立的。

隐私考虑

智能系统与那些可以执行“数据发现数据”的其他系统一样，要求非常密切地关注隐私权和保护公民自由。如何构建和部署这些下一代系统，以及如何使用管理策略（包括责任和监督）都值得人们密切关注和热烈讨论。一些核心问题包括：定义什么样的数据需要为可发现性构建索引，如何把这些数据整合在一起（比如，什么是关系？），什么是相关性，应该向谁揭示什么样的相关性，谁可以搜索索引，系统如何对未授权使用进行监视，以及如何检测错误和纠正错误。

幸运的是，基于目录的模型有很多关于加强隐私保护的很好的特征，包括：

·敦促多方之间分享更多的数据被传送更少的信息到更少的地方（卡片目录）所取代。

·可以通过一致的方式，对人们的搜索操作以及找到的结果以日志形式记录下来，这样有助于更好地实施责任制和监督制[1]。

·不同方之间的信息共享已经减少为只需要分享他们需要知道的东西（通过直分享必须分享的信息使得分享变少了）。

·可以使索引“匿名制”(aonymized)，这意味着无意的暴露甚至是那些索引中数据量很小的元数据的机会也大大降低[2]。

[1]防篡改的日志记录通常也被称为不可变的审计日志。Markle Foundation在关于国家安全，尤其是非透明的政府系统方面，发表了一

篇有趣的论文，可以浏览：

http：//www.markle.org/downloadable_assets/nstf_IAL_020906.pdf。

[2]更多关于“匿名制”的目录，可以在Jeff Jonas和John Karat所著的《Emergent Information Technologies and Enabling Policies for Counter-Terrorism》（编辑Robert L.Popp和John Yen，2006年Wiley-IEEE Press出版）的“Anonymized Semantic Directories.A Privacy-Enhancing Architecture for Enterprise Discovery”一章中找到(htp：//www.wiley.com/WileyCDA/WileyTitle/productCd-0471776157.html)。

结束语

“数据发现数据”系统决定了新的观察是如何和已知的事务相关，并检测出值得特别注意的相关性/洞察力。这种洞察力通常实时发生，因此能够做出实时反映。

这一新兴的技术将最终把一个组织和其他组织通过一系列的企业利益区别开，包括更好的识别机遇到风险估计。

数据发现数据很可能会成为下一个里程碑，下一代的高级分析机制会从中受益。这种新的模式将不可避免地带来更智能的系统的兴起，甚至可能还会推动“认知计算”(cgnitive computing)的发展。

第8章　实时的可移动数据

JudValeski

简介

长期以来，应用数据一直被“禁锢”。最初，这些数据被局限在世界各地“孤立”运行的系统中。CAT-5以太网络、IP、路由器、DNS和Sockets诞生之后，机器之间开始能够互相连接。机器间的互通使得数据可以在应用程序之间自由传输，最终这些数据也变得更加有意思。数据超出了其原始的应用程序范围，开始进入各种各样的新场景之中。一夜之间，随着互联网的兴起，客户端-服务器计算得到了迅速发展，那些“孤立”的计算机突然之间互相连接在了一起。以HTTP做为通信媒介和文本作为语言，数据“流动”了起来。

在过去大约15年里，主要的数据流可分为两类：色情和普通的消费商务（包括通过代理的广告），二者都很好地遵从20世纪90年代初期兴起的传统的由“Web浏览器”以及相应的服务器所构成的数据传送模型。最近，第三类数据——社交数据开始兴起。人们开始把自己的社交移到网络上，这创造了新的数据类型，以及新的不完全遵循基于浏览器模式的访问需求。社交数据就在这里，而且对它的需求正当其时。2009年1月14日的PEW报告《Adults and Social Network Websites》指出，18～34岁的在线成年人，50%以上有自己的社交网络个人信息[1]。

在本章中，我将讨论这一演变过程以及消费趋势信息即社交数据的内在问题的当前解决方案。虽然我们已经经历了跨越式发展，但这过程中丢失了一些基本的编程原则：事件驱动的体系结构和数据结构的可预

测性。通常，当前问题的解决方案就在眼前，而且“奥卡姆剃刀”(Ocam's razor)理论也表明这些解决方法通常很简单。真正通过180角度HTTP转换并采用POST方法，“实时”可以正式成为社交数据词汇的一部分。作为中介，Gnip(http：//www.gnip.com)在异构社交数据和消费应用之间架起了便捷的桥梁。

[1]http：//www.pewinternet.org/Reports/2009/Adults-and-Social-Network-Websites.aspx

前沿技术

当前的社交数据被“扭曲”来适应那些与最终用户所喜欢的行为需求并不完全匹配的传输和展现方式。当一个平台是经过很长时间逐步建立起来时，往往会发生这种情况；人们发现新的方式来利用平台，即使有时必须为此做出某些牺牲。详细说明当前框架的主要方面是描述这种“扭曲”了的杠杆作用的最佳方式。

传输

有很多方法可以把两个Socket连接起来，并在它们之中移动数据。但是，在可预见的将来，只存在一个可能：HTTP协议。总存在特定的情况，可以使用更高效的协议来优化该特定情况下的数据传输，但是HTTP协议仍然是无处不在的，虽然它比任何其他东西给地球和卫星的防火墙带来的漏洞都多。因此，这就是HTTP协议。如果有什么事不能通过HTTP协议来完成，那这件事就需要重新考虑一下了。为了说明这一点，我们具体看一下消费电子设备的情况。这些产品很多都很酷，但是只有那些很棒的设备才包含HTTP客户端（如iPhone或者一些基于Android的设备）。真正令人惊叹的设备甚至包含了HTTP服务器(Nkia的S60上可以运行HTTP服务器）。正如任何完善的规则都有例外，HTTP协议也不例外。它是无状态、重量级的，当你需要以快速、有状态形式完成一些任务时，这两个特征可能会成为障碍。以下协议是这些例外的例子。

可扩展消息处理现场协议(XPP)

AOL的即时通信产品是该领域在实时社会通信产品上的首次亮相，并且发展异常迅猛。人们想“聊天”。不久，人们就想通过P2P进行聊天，或者通过其他服务，XMPP就这样诞生了。开始的时候，作为一个标准的传输框架和用于描述社会交互、展现数据的标准格式，XMPP被拼凑成了一种通用的消息框架，使用范围广泛，包括从IM到通用消息工具PubSub。在2008年中期，Cisco收购了Jabber.com(XMPP安装商业化的领袖），网络路由器也将很快装配很小的、烧制了专用的XMPP处理程序的硅芯片，供全世界使用。

BitTorrent

当你的用例是高度定制时，最好能够把数据处理流程调成“最优”。砍掉这些琐碎的东西和多余的载荷，使得任意的客户端和服务器之间的数据传输通信只适用于客户端和服务器紧密耦合并且协作良好的情况。BitTorrent在分布式的一组节点上来回移动大量的二进制数据块的速度是异常惊人的，这是对高度定制的用例进行优化的一个很好的例子。

私有权/P2P

HTTP协议的一大不足是每个有效的请求/响应对包含了与之前相同的数据头集合，即使这对操作可能是某个特定的客户端和服务器的第n对的交互操作。如果能够在初始阶段管理好这些交互，使得数据通道除了位操作之外其他什么都不做，那岂不是更智能？确实如此。这个原因以及很多其他原因，驱动着“点到点”(Pint-2-Point或Peer-2-Peer)协议的很

多实现，这些实现每天传输着大量的数据。

选择HTTP协议作为传输方式，这将带来一个问题：它对应的格式是什么？当然，如果一个传输协议在过去几年中已经变得无处不在，这个过程中基于其传输的数据也必然已经被“净化”了。

格式

XML为进一步实现数据格式化打下了基础。数据应该基于哪一个层次格式来通信的问题基本上已经解决了。现在对由XML衍生出的数据格式的讨论是所有一切讨论的起始。虽然现在有很多开发人员觉得采用XML格式是理所当然的，但是值得注意的是这种数据格式非常臃肿。XML基于文本的格式，在定义上非常有描述性，因而它通常不是用于通信的最佳格式。每当我参与现代协议和格式的讨论时，我就想起AOL专有的FDO模型。FDO是二进制的、紧凑的以比特存储的协议/格式结合，为300波特的现代连接进行了优化。幸运的是，网络连接已经发展到今天，宽带速度已经成为规范。

如上所述，正如HTTP协议一样，XML的劣势也是它的优势。如果我们在过去15年学到了些东西，那便是无处不在的可读的协议和格式——我真的是这么认为。一种格式越是被人们所理解，它就越可以进一步走向创新之路。当调试程序的唯一方式是通过解密工具把数据转换成用大脑可以理解的东西时，人类创造性和通信之间就开始存在严重的障碍。“成功的数据”可以为程序员所理解，而“失败的数据”只能通过计算机来处理。注意：有效载荷格式和元数据格式（通常是XML)之间存在

区别。高清晰的视频数据（如最近的高清晰的轰动电影）占用了大量的物理空间，而且人们无法阅读它；我把这种数据称为有效载荷格式。描述性的信息，比如视频的元数据，是相对比较紧凑的。在网络中，通过有效载荷格式传输胜过元数据传输（只要问问你最喜欢的网络服务提供商它占多少比例就知道了），但是这种格式的数据在被计算机解码之前，相对显得呆板无趣。

接口

接口(AI)使事情变得开始有趣。一旦每个人都认识到大量的HTTP事务像传统的API调用一样在网络上无处不在，REST协议为我们把Web作为大分布式应用奠定了基础。Roy Fielding是HTTP1.1协议的贡献者之一，他教会我们如何像表示状态转换(Rpresentational State Transfer,REST)那样描述事物的思想框架。

开发者终于可以停止挣扎于使用CORBA和DCOM模式来把他们的逻辑和数据分布到联网的计算机上，转而借助于标准的HTTP和基于文本的工具来实现应用程序间在传统的Web应用层以下进行交互。

这导致你能够想到的几乎所有We b应用的API呈爆炸式发展，如图8-1所示。API成为每个产品经理必有的东西。这既是件好事，也是件坏事。从好的方面来看，成千上万的产品公开它们的数据，全世界都能够访问。Programmable Web(http：//www.programmableweb.com/）记录了好几百个这样的产品，而且随时可以使用。从坏的方面来看，很多API很仓促地开发出来，没有考虑到它们可能会如何被使用。其结果是由于

API调用程序根据性能或者功能特征，带来从WAN到各种可能满足或可能不满足你的期望的不同终端范围的API调用高潮。很少有Web API提供服务层协议(Srvice-Level Agreement,SLA)，如果真的有这样的API，其状况通常是很悲惨的。

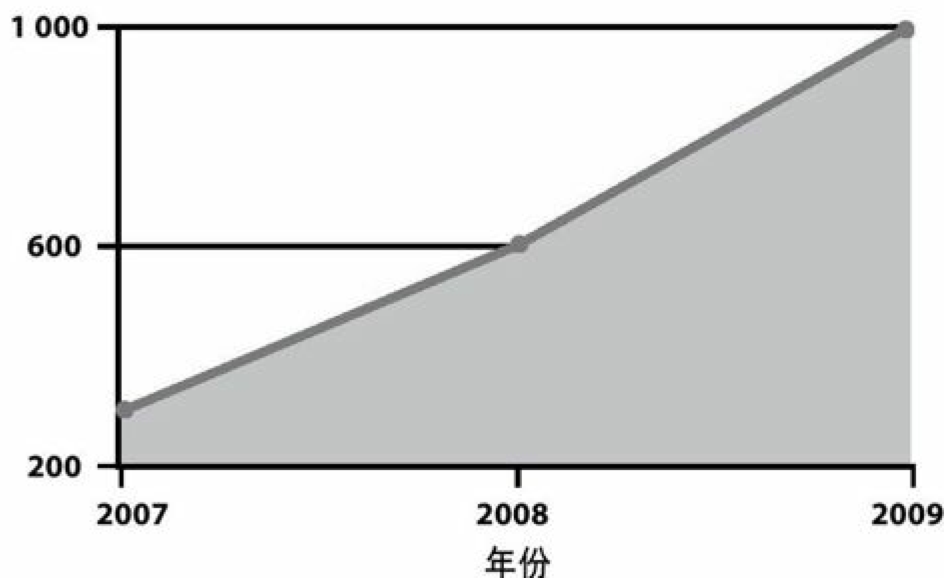


图　8-1：在过去几年公用的API数量增长情况，数据来源于Programmable Web网站

为了使某个应用的数据和功能能够通过API获取，不论是真正的还是假想的需求，都可以在网络上创造一些有趣的挑战。

如果查看今天的API在网络上是如何使用的，那么你会发现它们看起来是随意传输数据，且没有严格的SLA。在规模上，我们第一次有软件可以非常有规律地和其他软件进行集成。这种集成方式称为“混搭服务”(mshups)，因为它们把数据从各个服务中“混搭”在一起。越来越多的终端用户期望数据在他们的社交应用能够具有实时性，但是支持实时数

据传输的基础设施还远远不够丰富。在实现实时应用时，软件工程师无限复杂的创新性已经为应用提供实时通知产生了很多不同的模型（如“Comet”、“Web Hooks”，以及各种“PubSub”消息通知系统）。这些模型其中有些行之有效，有些则没有。不管你的方法是什么，解决方案在根本上没有真正的“魔术”；相反，饱受考验的框架通常都是在“幕后”一直发挥着作用。

在应用软件中有两个基本的时间流处理(tme-flow-processing)原语(Pimitives)：轮询(Plling)和事件(Eents)。应用程序根据时间来执行，而用户交互的应用程序需要用户或者另一个服务的输入。该输入通过轮询或者事件方式在应用中执行。值得注意的是你可以通过以上任意一种模式来实现，在计算机课堂上应该讲解到使用它们的一些绝妙技巧，我不在这里赘述。这里重要的是定义关于应用程序如何收集它的相关输入的更高层的概念。使用例子来说明表达这两种模型最简单。

轮询

轮询是通过应用程序不断询问中断是否有变化来实现的；如果检测到变化，程序会执行定义好的动作。一个常用的类比是酒吧服务员。想象一家酒吧，有1个服务员和10个顾客。如果这个服务员是轮询模式的服务员，他会不断地询问每个顾客他们是否要喝酒。在整个酒吧询问“要喝一杯吗？”“要喝一杯吗？”“要喝一杯吗？”“要喝一杯吗？”等。不时地，其中一个被询问的顾客会回答“来一杯”，则该服务员就先不询问了，给顾客倒一杯酒。轮询的方式实现起来美丽简单，但是通常这对

于客户端和服务器都是非常低效（正如类比所示）。

速率控制[1]

提供API为开发者进行访问的方式来展示你的在线服务，通常可以很容易实现；而基于应用程序构建一些框架并把数据连接起来就不是那么容易了。但是，一旦把API开放出来，你就无法控制这些API将会如何被其他应用程序使用。对程序的完全控制已经保护代码运行了几千年的时间。因此，构建一个大规模的“Web API”（即“Web服务”）需要付出慎重的考虑、熟练的管理和业务操作。

如果有人幸运地构建出一个令开发者很感兴趣的API，因此其轮询负载将变得很大，在应用上处理负荷增长的最简单的方式是通过IP或包头、通信层的访问，从而限制API的使用。虽然这种处理方式解决了伸缩性问题，但是它中断了为取悦用户而集成这个API的程序的处理流程。开发者不希望在他们的应用中构建“节流层”(Trottling Layer)，他们只是希望执行习惯性的调用，使得系统能够“工作即可”。实际上，在系统中解决数据访问问题需要花费的精力比基于IP的速率限制还要多（即“节流”），但是我们通常无法投入足够的时间来调研真正的解决方案。一些社交数据的应用已经投入精力来解决这个问题，他们的API能够承载高负荷访问（如Digg)。但是绝大多数应用还不具备这一特点。

做正确的事

然而，包含大量社交数据的流行的社交应用程序在基础设施上不应该要求有庞大的资金投入，也不必是具有很强横向扩展性的应用。这些

方面都会抑制产品的创新，耗费大量的资金。相反地，利用饱经时间考验的“事件”原语，简单地说是通过有序方式。大量规则说明了访问社交数据的90%以上的调用是不必要的。绝大多数时候，这些数据甚至是无法访问的。仍然以之前的酒吧服务员为例，当顾客回答“不喝酒”时，会由于被服务员不断地询问相同的问题而开始变得不耐烦。

为了更新某个用户拍的照片，不断地询问Flickr是一种资源浪费。相反地，如果当用户更新了他的照片，Flickr系统通知你的应用程序来调用，这种处理方式就会更高效，对用户更重要以及更及时。

每加仑0英里的效率

我将使用众所周知的“Flickr,Friendfeed的例子”来强调在获取社交数据时，使用轮询方式的低效性。总体来说，社交数据是由用户生成的。这意味着依赖于利用社交数据的社交应用是基于人们的行为，而不是更容易预测和管理的计算机的行为。当我们开发的软件的价值完全是人类活动的某个功能时，事情开始变得有趣。这个例子很有趣，因为发布者(Fickr)和消费者(Fiendfeed)都对用户的笔记留言进行比较，这能够对通过轮询方式异步传输的“外部调用”(ofbox)的本质问题产生有价值的洞察。

在2008年7月，Even Henshaw-Plath和Kellan Elliot-McCrea在俄勒冈州波特兰(Prtland)市组织的OSCON(Open Source Convention,O'Reilly的开源集会）会议上发表了演讲“Beyond REST?”(htp：//www.slideshare.net/kellan/beyond-rest)。在该演讲中，他

们透露为了确定Friendfeed的4.6万的用户账号在最近24小时内是否上传相片，Friendfeed每天会请求Flickr的API 290万次。而它的4.6万的用户，其中仅仅7000个用户在这期间内会访问Flickr，可能会上传照片。

一条著名的社交应用规则，称为“1-9-90”规则，在过去几年中开始兴起。其要旨是，如果你考虑社交应用100%的用户基础，其中1%的用户对核心数据有贡献（如“上传一张照片”），9%的用户参与贡献数据（如“把一张照片标记为最喜欢的”），而90%的用户仅仅浏览这些数据（如“视图照片”）。如果保守地把该规则应用于Flickr/Friendfeed范例中，则可以说明在某一天有7000个用户访问了Flickr，有700个用户会上传照片。这意味着Friendfeed请求Flickr接口290万次，只是为了知道他们的700个用户确实做了某些更新。这种方式“命中”的最高比率是0.02%，即4000次请求发现1条更新。

把Flickr/Friendfeed例子和酒吧服务员的类比结合起来，意味着轮询方式的酒吧服务员为了使得顾客需要酒时可以实时得到，需要询问这位顾客4000次“要喝一杯吗？”，他才会回答1次“来一杯”。

一方面，应用程序调用本地的“内部调用”(O-box)API来“吸纳”这种低效性。另一方面，应用程序执行Web API调用的效率也非常低下。如果你根据当前所有的社交应用来考虑这个例子，其代价确实会非常令人震惊。

事件驱动型的软件可以驱动应用程序达到实时性，而社交数据在本质上就需要实时通信。通过在线银行软件查看某张支票账户清除了我的

支票账户的操作，“拖延”一两天才能看到是可以容忍的；但是一个年轻人要看到他最好的朋友在春天拍摄的照片的请求，需要被立即处理。

事件

事件驱动型的架构和轮询驱动型不同，它采取了不同的解决方案。和酒吧服务员不断地询问顾客们是否需要喝酒不同，这种模型的酒吧服务员仅仅站在吧台后面，等待顾客告诉说他们想要喝酒。虽然事件处理通常更有效，但是考虑到具体的“事件”概念本身，它确实要求一些额外的代价。事件需要被消除、捕获；事件处理框架必须存在，这导致了事件处理方式的复杂性。轮询方式可以简单地通过线性、过程化的循环来完成，而这种简单性也正是它被过度滥用的一个关键原因。

从侏罗纪时期开始，轮询和事件这两种执行流规范奠定了软件开发的基础。客户端和服务器软件都依赖这两种模式来控制执行流以及应用的行为。操作系统通过恰当地运用它们，提供流畅的用户输入和交互体验。作为一个开发人员，你选择哪一种模型通常取决于你所使用的开发语言，或者你正在使用的库或框架的功能。虽然这两者的性能特征在本地、内部API调用的应用中通常区别不大，但是当执行远程、外部API调用的操作时，其性能特征可能导致应用程序瘫痪。注意，当处理用户界面和图形渲染时，本地性能的区别对于应用是至关重要的。

一旦程序员开始大量注入远程API调用到本地应用程序，轮询和事件处理方式之间的区别变得非常明显。本地I/O调用和远程I/O调用之间的区别呈指数增长。所有通过磁盘、内存和芯片接口来加快应用程序的工

作变得毫无意义。图8-2说明了本地数据访问和远程数据访问之间的I/O性能的相对区别。为了简单起见，我已经隐含地说明了固有的IP连接断开延迟，并且作为带宽/吞吐量功能来说明这一点。然而，如果你真正考虑实际的协议交互作为全部数据传输代价的功能，延迟问题会变得更加严重。

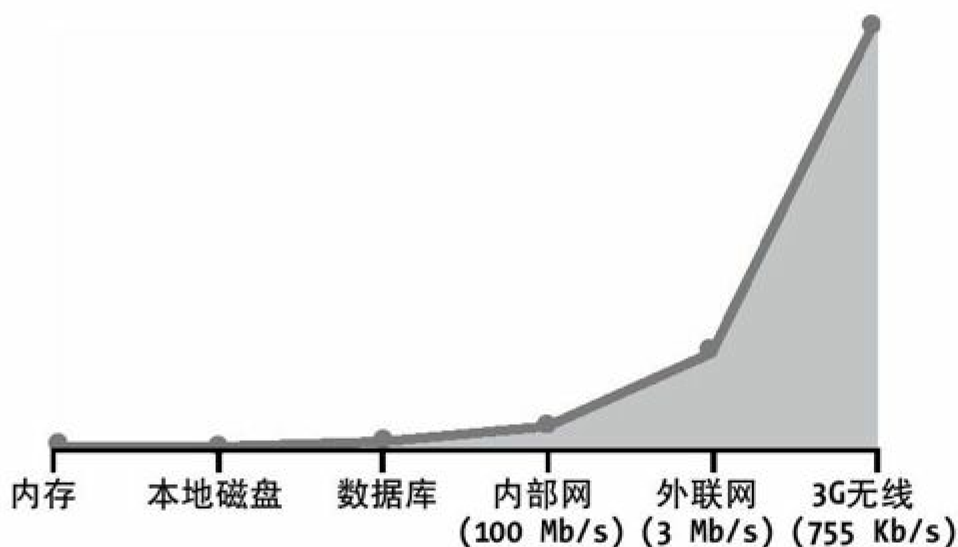


图　8-2：文件描述符在各种连接类型上的操作的相对吞吐量

正如你所看到的，差异非常明显。在远程数据访问次数远多于本地数据访问的极端情况下，轮询和事件模式在驱动应用时所产生的延迟的差别会更加明显。当你的软件为了采取某些措施而需要知道在网络中某个服务是否发生变化时，你往往没有足够的时间来执行低效的轮询。如果酒吧服务员问了4000次的“要喝一杯吗？”，却只能得到顾客1次的肯定回答，当服务员需要问遍城镇里的所有顾客时，浪费在查询上的时间都要爆炸了。轮询和事件之间各有千秋，但是只有一种模型在Web上易

于实现：轮询。基于HTTP/REST的客户端-服务器编程从来没有为Web应用开发提供正式的事件驱动框架。结果导致Web充满了低时效性的社交数据驱动的应用，社交数据的时效性特征让这一问题更为突出。如果我的朋友明天可能在城镇，我可不想自己知道时为时已晚。我们希望应用能够“实时”就像构建在要倒塌的扑克牌堆砌的房子那样，而我们看到墙变得越来越高。

HTML 5事件

值得注意的是HTML5规范突出了事件模型和在高层上支持双向通信功能的“Web Socket”框架。XMLHttpRequest已经把GUI Web应用提升到一个新的层次，我猜想“Web Sockets”将对今后几年的Web应用开发产生重大影响。允许基于浏览器的应用以更类似于socket的方式来访问数据必将带来好的结果。

WAN规模的事件

分布式事件处理和通知不是一个新问题，而它的解决方案也不是。事实上，企业空间已经生成了无比强大和高效的“消息总线”的解决方案，尽管这是专利的。Tibco公司作为企业消息总线解决方案的领先提供商而广为人知。然而，企业消息总线的关注重点在于它们传输的数据的临界值，而非通过各种各样的连接堆栈方能实现与多样化的终端间进行连接。股票交易需要确保传输成功并保护隐私。要达到这种程度需要付出较高的代价，该代价转化为了高额的牌照费，但Internet上，这一转化进行得不是很好。社交数据是通过对终端用户免费的应用创建和消费

的。因此，各企业公司提供的昂贵的、私有的和难以集成的消息总线解决方案在Web上不受欢迎。

利用生成社交数据的技术来提供事件驱动通知机制的解决方案，通常正是基于该系统已有的技术。可以使用HTTP POST方式来推动整个WAN范围内任意服务之间的事件，使用标准格式(XL)，在执行控制流过程中的延迟可以被关闭。Jeff Lindsay在宣传HTTP POST的事件传输模型时给出了一个生动的比喻，并由此获得了绰号“Web Hooks”(htp：//blog.webhooks.org/）。再强调一次，HTTP不是最优化的传输事件协议，XML也不是最优格式，但是这二者提供了最丰富、实用的机制来解决服务器之间实时通信的问题。

如果社交数据变得更加隐蔽，有可能发生以下两件事之一：一是我们作为消费者，将继续忍受，而我们的行为模式和期望将会更符合低级模型；二是我们停止使用该产品，因为它们的使用价值已经下降到不足以抵消使用所需要忍受的烦恼。前者有很多先例(Bta与VHS)，因为我们通常接受我们想要解决的问题的非最佳解决方案。后者每天都发生，因为发展迅猛的工业和产品通常由于时间和最后的优化因素而崩溃。在一个框架变成标准之前，中间的事件网关可以通过众多协议和格式代为转发事件和数据的方式，起到连通数据发布者和阅读者的桥梁作用。如果事件机制的实现可以做到成本低且通用，它将可以消除社交数据的破坏性的延迟。“Web规模”的实时社交数据的普遍应用依赖于事件驱动模型。

Gnip项目[2]是基于以下理念成立的：围绕社交数据消费的事件驱动架构是提供实时访问模式的唯一方式。然而，由于轮询方式还会存在，Gnip仍然选择轮询方式作为社交数据交互的首选模型。

单一的API集成通常简单明了。然而，当你要集成很多不同的终端时，便开始出现效率低下的问题。Gnip为它的终端数据消费者实现了多API集成的工作，这些API最后只需要通过唯一点就可以完成集成Gnip。

[1]在计算机网络中，速率控制常用在一个网络接口上，用于控制发送和接收数据的通信速率。流量小于或等于指定速率的数据会被发送，而流量若是超过了指定速率，则被丢弃或延迟发送。

[2]想了解更多信息，可访问其网站：http：//www.gnip.com/。

社交数据规范化

假定我们通过HTTP POST来处理广域网范围内的事件的方式——Web Hooks——解决了数据访问延迟问题。这种方式只是解决了通用的API访问问题，而不是数据本身的多样化特征问题。XML提供了数据的结构化特性，但是没有考虑到通用性。当今的社交数据聚集应用和它们要集成的各种类型的社交数据的API都碰上了“一次性理解”的难题。理解从一个特定API获取到的数据的结构需要付出的代价很高——太高了。虽然多路复用网关协议已经存在了很长时间，但是通常对于加强来自稀疏数据源的通用的、规范化的数据，只存在严格的XML转换翻译器。遗憾的是，严格的数据解析方式很少能够正常工作，因为实际上各种服务集创建的数据是非常多样化的。人们对标准、编码和转义序列的理解都是不同的。此外，创建XML用于消费的软件不可避免地存在bug，这导致输出格式简陋，对其使用变得进一步复杂化。

我们从严格的HTML解析Web浏览器中明白，遗憾的是，标准并不能使那些和标准保持一致的软件能够准确无误地解析这些非完美的数据格式。实际情况是标准有不同的解析方式，软件存在bug，而且最强大的事实标准是用户已经正在做的事。人类的强大永远都不能被否认。

如果你曾经花一些时间从各种社交数据源看这些数据，你会注意到这些数据变得看起来都差不多。虽然要解释的通用性对人类而言很明确，但对机器来说却很困难；需要人工编辑指南给出从数据集A到数据

集B的映射。

从两种不同的“社交标签”服务考虑以下两个XML例子。虽然它们显然都是XML，但是其“书签”的表示方式差别迥异，而且它们都为终端用户提供相似的服务。

来自Delicious网站：

```
<item>
<title>Fractals derived from Newton-Raphson iteration</title>
<pubDate>Mon，19 Jan 200920：02：05+0000</pubDate>
<guid isPermaLink="false">
http：//delicious.com/url/7549fded443f#joe</guid>
<link>http：//www.chiark.greenend.org.uk/～sgtatham/newton/</link
>
<dc:creator>iacovibus</dc:creator>
<comments>http：//delicious.com/url/7549fded443f</comments>
<wfw:commentRss>http：//feeds.delicious.com/v2/rss/url/a
</wfw:commentRss>
<source url="http：//feeds.delicious.com/v2/rss/joe">joe's
bookmarks</source>
<category domain="http：//delicious.com/joe/">mathematics
</category>
<category domain="http：//delicious.com/joe/">newton-raphson
</category>
<category domain="http：//delicious.com/joe/">fractals</category
>
<category domain="http：//delicious.com/joe/">iteration</category
>
</item>
```

来自givealink.org网站：

```
<item>
<title>Bus slams into shop houses after driver collapses behind
wheel</title>
<link>http：//www.thaivisa.com/forum/Bus-Slams-Shop-Coll-
t198228.html</link>
<description>Bus slams into shop collapses behind wheel
</description>
</item>
```

当今，有成千上万的服务通过API和Feeds暴露其用户生成的内容(Uer-Generated Content,UGC)，对其结构和内容进行规范化，这样开发人员可以期望通用性优先级变得更高。DiSo项目是把整个API范围内相关成员引入该方面的主要催化剂，用于“净化”更多可使用的社交数据参见http：//diso-project.org/wiki/activity-streams。

作为数据生成者和数据使用者之间的媒体，Gnip处于一个得天独厚的地位来把社交数据的意义翻译和规范化到传统的理解和结构中。利用整个行业范围内需要更易于消费的社交数据提供集体输入和知识，Gnip作为“漏斗”，接收的是很多不同的协议和格式，而输出的是一致、异构的数据，并提供了对数据更便捷的访问方式。

数据的商业价值

关于开源软件是好是坏的争论基本结束。企业何时对其软件或者是部分软件进行开源，何时不能开源，有很清晰的界限。总体来说，如果你的软件享有“独一无二”的知识产权，把你和外界的其他方面分离开，你应该考虑不把它开源。否则，它就可以作为开源，这样开源社区可以在代码中涵盖其经验和专业知识，为全世界作出贡献。遗憾的是，软件开源而言的数据相对决策框架还不够成熟。API的爆炸使得数据发布商措手不及，因此，他们对数据价值的理解变得糊涂得一文不值。

一些传统的内容发布商已经强化了其数据的价值。提供免费的周刊来提供内容，包含广告商支持内容（评论）和产品（杂志）的生成。一些贸易杂志收取订阅金，不加入广告模式。Internet上的传统的媒体/内

容发布商主要采取广告模式来支持他们的数据（内容）的分布。然而，通过API访问社交数据没有成熟的模型。提供商是否需要为数据本身的访问收费，还是一切都免费？除非你是一个“freegan”（反消费者），否则以上问题没有一个有明确的答案。除非谈及的数据存在内在价值，否则对其评价变得非常困难和曲折。如果你作为数据使用者，为了购买微博客消息，是否应该花费0.01美元？这个消息是否应该获得许可，或者访问方式是否应该被租用？数据项的价值和对它在转换处理之间的区别开始变得很明显。用户在发布商的服务上真正去生成内容，是否应该获得其中收益的一部分？从刚开始，谁是真正的数据拥有者？正如你所见的，这里充斥着很多这样的问题，它们没有明显的答案。

当出现了社交数据，在该行业中，一些发布商会声称他们的数据是无价的，而另一些则认为数据是免费的。数据使用者已经潜意识地相信绝大多数的数据是免费的。因此，如果真的对社交数据进行标价，他们便会发展成价值链切实的一部分。

公共的与私有的

有些数据被认为是公共的，而其他数据则被认为是私有的。公共和私有这二者之间的区别是通过提供数据的服务来定义的，而且通常是通过给定服务的《服务条款》来概括的。虽然对这二者的社会理解及其本身是很有意思的话题，但是我在这部分将考虑访问这二者的技术上的一些隐式问题。

让我们从两个当中更简单的公共数据说起。访问公共数据相对来说

比较简单。今天绝大多数的数据API是通过未授权的REST接口来访问的。这意味着在网络上它们和其他的URL并没有什么区别，它们可以很容易被任意用户和应用程序利用。访问这些API终端需要很少的身份验证形式，如果有的话，而且对API用户的授权通常是通过不透明的方式在幕后实现处理的。两种更丰富的验证方式是使用HTTP Basic-Auth和嵌入在URL中的“API关键字”。这两种方式都允许API服务基于身份验证证书来控制对数据和API功能的访问。令人惊讶的是，这两种方式和HTTP都能工作很好。不考虑对公共数据的API访问是否需要任何身份验证，底层的社交数据在绝大多数情况下仍然是“公共的”。

私有数据带来了很多麻烦。由于越来越多的应用开始利用通常被认为“私有”的其他应用的数据，这需要控制终端用户的访问。数据提供商还使用各种技术，在访问权限控制中叠加上自己的观点意见。这些技术都使得私有数据的聚集访问方式变得异常复杂。

保证终端用户数据在存储角度是受到保护的，这仅仅是挑战的一部分。存储冗余和加密技术允许应用“蓬勃发展”，通常不需要考虑用户数据的丢失和折中方案。实际上，用户对应用程序很放心，为了允许更深层的API集成，他们免费提交用户名和密码到第三方应用程序。这种做法实际上非常不安全，但是它说明了用户为了获得他们期望的功能，甘愿牺牲其隐私信息。有两种有效可选的方案用于分享私人登录证书：OAuth和Blind URL。

OAuth(http：//oauth. net/）提供了简单的解决方案来允许各种服务之

间进行交互，所有这些交互都给终端用户在他们的信息上提供了最终的权限控制权，而不需要和第三方透露他们的用户名和密码。交互程序把用户交给其期望的集成点，询问用户是否允许集成服务所请求的访问权限控制。如果终端用户赞成该交互，两个服务之间就共享一个令牌(Tken)，而且用户也可以选择撤销服务的访问权限许可。

Blind URL是在服务和用户之间共享信息的更简单的方式，但是它们在技术上还不够“安全”，因此它们通常都没有什么可信度。Flickr的“Guest Pass”共享功能能够巧妙地利用盲目的URL来允许用户和那些没有Flickr账号的用户分享“私有”的照片，因此，这些用户不属于Flickr用户的“朋友”或者“家族”。Blind URL为了使用服务来获取URL的使用许可，它不需要用户做出任何额外的工作，但是它们从技术上能够猜测出来，这会带来私有信息泄露的隐患。

Gnip当前只处理公共的数据。对于公共的和私有的边界访问控制必须提供给终端用户，而且当讨论中间基础设施时，它并非总是很清晰的。例如，你不知道也不关心公司采用什么样的通信路由服务信用卡来保证金融交易，而对于谁能够访问你的假期照片的访问权限的控制却是至关重要的。Gnip正在努力在公共数据和私有数据之间画线，以及它最终如何为私有数据提供如公有数据一样的服务支持。

结束语：通过Gnip思考

每天“汹涌”的数据“浪潮”把更多的异构API“拍上海岸”，而在这些API中，必须保证数据的一致性。作为面向消息的中间件服务，Gnip公司承诺使用中间数据代理的方式来“传送Web数据”。这意味着该数据发布商期望通过很多内部的传输协议来分享这些数据，Gnip从发布商获取规范化的数据，并实时地传送给Gnip用户（从消息接收到重播时间延迟低于60秒）。不考虑内部的数据格式不一致，Gnip把数据“清洗”并“规范化”成标准的格式作为服务提供给Gnip用户。其结果是只使用Gnip的单点集成，就可以为其他应用提供一致的、格式化的数据。

你可以通过Gnip一个接口来访问很多的社交数据API，且都是“实时”地。Gnip的框架主要采用之前所述的事件模型，为了促进网络和应用之间总体上更高效的数据流。然而，它也支持轮询方式。把基于轮询的应用切换到基于事件的模式来最大化利用Gnip提供的规范化的数据的优点，而通过手工处理将很麻烦。如果不是同步，那么完全采取尚待完善的发布/订阅框架和格式一致性标准，采取中间代理是很有必要的。

当前的基础设施在推进事情进展方面已经做了很多工作，而且网上贸易已经蓬勃发展。然而，不断改变用例和终端用户需求使得和当前框架保持一致会使我们受到一些局限。实时社交数据的需求要求在构建我们的应用的底层控制流上做出修改。我很期望看到基于事件的架构可以扩展到整个Web。

第9章　探寻Deep Web

Alon Halevy和Jayant Madhaven

什么是Deep Web

“Deep Web”（深网）指的是隐藏在HTML表单后的Web内容。为了获取这些内容，用户必须提交包含有效值的表单。例如图9-1的商店定位表单，搜索邮编为94043的商店，可以得到一个包含了商店列表的Web页面。该结果页面即为Deep Web的Web页面的一个例子。

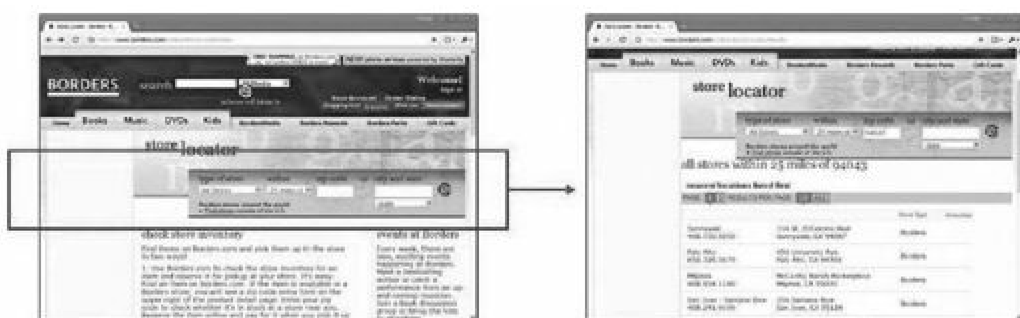


图 9-1：Borders商店定位表单和提交特定表单得到的Deep Web结果页（见彩图21）

Deep Web被公认为是搜索引擎在覆盖率上存在的一个很大空白。这是因为搜索引擎通过Web爬虫(cawler)来发现页面，并在索引中包含这些页面，从传统意义上来说，这些Web爬虫仅仅依赖于Web页面的超链接来发现新的Web内容。它们缺乏自动提交表单的能力，因此在表单后的Web页面无法被搜索引擎索引。包含表单的页面通常对表单后的页面内容包含很少；因此，对于普通的用户，只有当他们已经知道相应的HTML表单或者搜索引擎以某种方式将他们引导至该表单，这些用户才能够发现这些Deep Web内容。事实上，Deep Web（也称“不可见网

络”(Ivisible Web)或者“暗网”(Hdden Web)）的名字正是基于以下观察——Web用户无法通过搜索引擎简单地访问这些内容。

很多文章认为Deep Web拥有的数据比现在可搜索的WWW要多得多(Brgman 2001，He 2007，Raghavan 2001）。我们最近的研究(Mdhavan 2007）估计，存在几千万包含潜在有用的deep-web内容的HTML表单，而且Deep Web涵盖的范围跨越了每个可以想象的领域。这些流行的领域包括二手车销售、房地产清单、公寓租赁、求职、产品和食品食谱。有很多表单可以访问政府或者公共部门信息，比如法律法规、法院裁决、环境报告等。还有些表单可以让用户搜索更加深层的内容，如绿荫树、去公园的打车费、马雕像等。

当考虑Deep Web时，我们必须记住HTML表单适用于Web中的各种任务，但不是所有表单都可以访问deep-web内容。例如，要求输入用户名和密码的登录表单、提交用户输入到论坛和博客的反馈表单、执行购买的购物车等，这些表单需要获取用户的私有信息或者导致后台状态改变，因此不属于Deep Web的一部分。相反，我们主要考虑那些允许用户匿名搜索信息的表单。

由于deep-web内容的本质特征和数量规模，很自然地，搜索引擎会希望在它们的Web索引中包含这些内容。这样，搜索引擎才可以向用户展现这些新的内容，反之亦然。因此，无论在学术领域还是工业领域，人们对如何提供访问deep-web内容的问题都很感兴趣。但是，很大一部分的研究和技术将重点放在了如何在某些狭窄的领域来解决这个问题。

这种做法最显著代表是垂直搜索引擎，每个都是专注于某个狭窄领域的内容。例如，二手车搜索和求职搜索网站允许用户从单一门户网站搜索很多底层的站点。这种策略虽然可以访问一部分Deep Web内容，但是其可达的范围非常有限，并且忽略了大量不适合于这些狭义领域的基于表单的网站。

我们的目标是为通用搜索引擎用户提供Deep Web访问方式。从搜索引擎的角度，我们希望包含很大规模的Deep Web内容，即能够访问尽可能多的上千万的HTML表单。因此，我们需要一个解决方案可以在所有可能的语言和领域工作，而且不需要人工监督，即一个完全自动化的策略。从deep-web站点主机(hst)角度考虑，该策略不应该过度使用站点宿主的资源；换句话说，该策略只应该驱动那些真正相关的用户流量到该网站。

在这一章，我们将概述一个满足上述标准的解决方案。该方案名称是探寻(srfacing)，对于每个HTML表单，预计算一组查询，这组查询可能能够从深层站点中返回有用的内容。从每个预计算的查询中汇集URL，再把这些URL插入到搜索引擎的索引。从高层上看，我们的方法解决了两个挑战：一是在向一个表单提交查询时，决定需要填写哪些表单输入框；二是找到合适的值来填写这些输入框。在核心上，该方法依赖于通过智能选择的样本查询来探测HTML表单，然后分析检索到的Web页面的区别。该策略虽然简单优雅，却是非常高效的，而且可以有效地为Web搜索用户提供很大一部分Deep Web内容。我们的搜寻策略可

以应用于Web中任何使用get方法（在本章后面会概述）的HTML表单。这主要是为了排除一些我们不希望爬虫去抓取的表单，比如需要用户信息的表单或者是产品购买结果表单，这些表单通常使用post方法。

我们的deep-web探寻系统已经部署到G公司的搜索引擎上。该系统成功地抓取了几百个领域下的数百万个站点。目前，在G公司的网站上，每秒有超过1000个的查询，在首页结果中能够看到一条来自Deep Web的搜寻结果。总体来说，搜索引擎用户发现这些结果和普通的搜索结果一样有用。关于该解决方案更详细的探讨和实验分析请参考(Mdhavan 2008）和(Mdhavan 2009）。

在本章的剩余部分，我们首先探讨访问deep-web站点的其他可选方案；其次描述思考该问题的概念模型；然后描述预测一个表单中有用的输入框（以及输入框组合）以及预测适合于这些输入框所需的合适值的策略；最后总结并评论我们在方案部署方面的经验。

提供Deep Web访问的其他可选方案

访问deep-web内容有两种通用的方法。第一种方法是为特定领域（如汽车、书或者房地产）创建中间协调器(mdiator)，这种方法被垂直搜索引擎广泛采用。在这种方法中，我们先创建一个主(mster)表单（即作为中间协调器），然后在每个表单和该中间协调器之间创建语义映射。对于每个中间协调器上的查询，基于一些预计算的表单摘要信息选择相关的底层表单；通过语义映射在每个表单上构建查询；接着从每个选择的表单中返回内容，然后组合这些内容呈现给用户。从高层上看，该方法和现代提供购物对比的门户网站的实现思想很相似，这些门户网站通过使用Web服务，返回很多底层站点的供应信息。

由于垂直搜索专注于一个领域内的同类表单的收集，因此这种方法可以适合于垂直搜索，但它不适合于通用的搜索引擎。首先，构建和维护很多不同的中间协调器和映射的人工代价很高。其次，识别对搜索引擎关键字查询最相关的表单是非常具有挑战性的。只有小部分的表单需要识别，否则，底层表单接收到的用户流量可能超过了它们的处理能力。为了实现相关表单的识别，极端情况下，表单的摘要信息可能需要和底层内容几乎一样大。最后，即更基础的原因是，Web上的数据可以是方方面面，无法给领域的边界做出清晰的定义。因而，为Web创建一个中间协调器将是一项宏伟的挑战，而且该中间协调器需要支持100多种语言。因此，当目标是覆盖很多领域上的大量表单时，这种方法是不

可行的。

第二种方法是探寻，为任何感兴趣的HTML表单预先计算最相关的表单提交。每个表单提交生成一个唯一的URL，该URL可以像任何其他的HTML页面一样被搜索引擎索引。这种方法能够充分利用现有的搜索引擎基础设施。此外，它可以将deep-web页面无缝地融合到Web搜索中，也就是说，这些deep-web页面可以被直接插入到结果页的排序列表中来响应搜索查询请求。此外，如果用户基于页面摘要(sippet)认为搜索结果是相关的，当点击该搜索结果时，就可以给deep-web内容带来用户流量。

实现探寻方法[1]主要面临两个挑战。第一，HTML表单通常含有多个输入框，因此，对所有输入框的所有可能值的整个笛卡儿积进行枚举的简单策略，会导致生成海量的URL。Web爬虫需要抓取每个生成的URL，抓取太多的URL会耗尽Web爬虫的资源，而且很可能会对这些HTML表单的宿主Web服务器带来无法承受的负荷。此外，当笛卡儿积非常大时，很可能会有大量的结果页面是空的，因而从索引角度考虑，这些结果页是无用的。例如，在Cars.com上的某个查询表单有5个输入框，这些输入框的笛卡儿积生成的URL超过2.4亿个，而实际上该网站只有65辆车待售(http：//www.cars.com)。毫不奇怪，对于这些URL，大量的表单提交将无法得到任何记录，因此对于搜索引擎是无用的。

第二，HTML表单通常包含文本输入框，而且可能需要输入合适的值才能够返回检索结果。文本输入框可以有两种类型：一是接受任意输

入值的通用输入框，如关键字搜索框，二是只接受某个特定集合中的值键入的输入框，图9-1所示的是商店定位的邮政编码输入框。

本章的剩余部分所描述的解决方案采用如下策略来解决这些挑战。首先，我们设计了称之为“信息度测试”(iformativeness test)的测评方式，用来评估查询模板，即表单输入框的组合。对于任何模板，我们通过不同集合的值来探测表单中的模板里的输入框，并且检查我们得到的HTML页面之间是否有显著不同。生成的结果页面显著不同的模板被认为是作为探寻的好的候选模板。其次，为了识别适合于探寻的查询模板，我们设计了一个算法，它可以高效地遍历查询模板的空间。该算法权衡了生成更少的URL和达到网站内容的高覆盖率。最后，我们设计了算法来预测文本输入框的合适的输入值。该算法扩展了之前为通用文本输入选择关键字的算法(Brbosa 2004，Ntoulas 2005），采用富信息量测试方式来表示从通用的数据类型中识别键入的输入项的值，比如邮政编码、价格和日期。

HTML表单处理的基础

这里我们只是简要地概述了表单处理。更多的信息可以通过HTML表单规范来获取(htp：//www.w3.org/TR/html401/interact/forms.html)。

HTML表单是由form标签定义的（如例9-1）。参数action表示当表单提交时，需要执行查询处理的服务器。表单可以有若干个输入控件，每个控件通过input标签定义。输入控件可以有很多类型，其中最重要的是文本框(txt box)、下拉选择框(slect menu，通过分离的select标签定

义）、复选框(ceckbox)、单选按钮(rdio button)和提交按钮(sbmit button)。每个输入项都有一个name，它通常不是用户可以直接在HTML页面上看到的名字。用户可以选择以下方式选择输入值：在文本框中输入任意的关键字，或在下拉选择框、复选框和单选按钮中选择预定义的选项。此外，存在隐藏输入框(hdden input)，它的值是固定的且对于和表单交互的用户是不可见的。隐藏输入框通常用于为表单提交提供一些额外的上下文信息（如该表单的来源站点）。在这一章中，我们集中研究表单中的选择按钮和文本输入框。复选框、单选框的处理方式和下拉选择框相同。

例9-1：包含让用户搜索工作的表单的HTML代码

```
  <form action="http：//jobs.com/find"method="get">
  <input type="hidden"name="src"value="hp"/>
  Keywords：<input type="text"name="kw"/>
  State：<select name="st"><option value="Any"/><option
value="AK"/>
  <option value="AL"/>……</select>
  Sort By：<select name="sort"><option value="salary"/>
  <option value="startdate"/>……</select>
  <input type="submit"name="s"value="go"/>
  </form>
```

当提交一个表单时，浏览器会通过get或者post方法，发送包含输入值的HTTP请求到服务器。如果采用get方法，参数会附加到action参数值后，在HTTP请求中，作为URL的一部分（如例9-1的http：//jobs.com/fnd?src=hp＆kw=chef＆st=Any＆sort=salary＆s=go)。如果采用post方法，参数会在HTTP请求体(bdy)中发送，URL仅仅包含

action（如例9-1的http：//jobs.com/find)。因此，使用get方法从表单中得到的URL是唯一的（依赖于提交的参数值），但采用post方法得到的URL不是唯一的。

因为搜索引擎是基于Web页面的URL来识别它们，从post得到的结果页面无法区分URL，因此不能直接插入到搜索引擎的索引中。此外，按照HTML规范，当表单提交导致状态变化或者产生副作用(sde effect)（如购物车、旅行预定和登录），必须使用post表单。正如之前描述的，这样的站点通常本质上没有信息。由于这些原因，我们把注意力限制在get表单，这种表单可以生成更适合于Web搜索的内容。

任何需要输入个人信息的表单都必须去除，举个例子，通过过滤任何包含密码输入框以及任何如用户名、登录等通常都和个人信息相关的单词。同样，只是简单地记录用户反馈或者评论的表单也可以通过以下方式去除：忽略那些包含有文本区域输入框的表单。

最后一点，我们提出的方案无法处理Javascript事件（正如本章所描述的）。表单和输入控件可以包含onselect、onclick和onsubmit属性，这些属性可以包含任意的Javascript代码。处理这些事件意味着要模拟Javascript在所有可能事件下的执行。原则上，我们可以利用Javascript引擎如SpiderMonkey和V8（见本章的“参考文献”），通过利用Web浏览器为我们执行表单提交，从而扩展我们的算法来处理这些表单。但是这种模拟的处理代价很高，因而其挑战在于能够快速地识别那些很可能会生成deep-web内容的Javascript表单，而且可以把这些deep-web内容添加到

搜索引擎索引中，也就是说，最后的那个Web页面有一个诸如get请求一样的“幂等URL”(iempotent URL)。

查询和查询模板

我们可以把表单理解为允许用户向后台数据库提交查询的接口。每个表单提交都是一个查询，这个查询的输入是表单的输入框中提供的值，返回的是数据库的记录子集。该查询属于由表单输入项和其值约束的限制性语言(rstricted language)。此外，表单站点的内容在查询之前是未知的。因此，为探寻选择表单提交的问题实质上是在内容未知的数据库中的限制性语言中选择一组查询。

选择正确的一组查询的挑战在于表单输入项的模糊性质。尤其是，输入项可以有两种类型。第一，选择输入项是在数据库记录上增加选择条件，如例9-1中的kw（在求职描述中的关键字(kyword)）和st（状态(sate)）。选择输入项的值可以从一个预定义的列表（通过选择菜单）中获取，或者在文本输入项中输入。文本输入项可能只接受特定类型的值，但其类型我们通常是无法知道的。选择输入项通常可以分配一个可以匹配数据库中所有记录的通配符值。对于选择菜单，该通配符必须是该菜单的一个选项，如输入状态有Any这个选项值；对于文本输入项，其通配符是空字符串。

第二，存在表示层输入项，它们不影响记录的选择，而只是控制表示方面，比如结果页面的排序或HTML页面布局，如例9-1中的排序。对选择输入项和表示层输入项进行区分是探寻算法需要解决的一个挑战。

形式上，假设我们需要使用SQL来提交查询。我们可以把表单站点的内容建模为只包含一个单表D的数据库，该表包含m个属性。每个表单提交可以看做一个如下查询：select*from D where P，其中P表示通过选择输入项来表示的选择谓词。

例如，假设例9-1的表单是用于在job表Jobs(position、city、state、desc)上提交查询，提交请求返回在美国加州的厨师岗位的检索，其对应的查询应该是：select*from Jobs where state='CA'and desc like'%chef%'。注意一点，我们这里假定在例9-1的其他输入项是表示层输入项。

探寻的问题本质上是选择一组较优的查询（表单提交）。但是，对每个表单提交的属性进行推测是不现实甚至不可行的。因为一个表单可能对应几百万种不同方式的表单提交，分别测试每个表单提交可能会耗尽底层站点的资源。为了推测表单提交的集合，我们定义了查询模板。查询模板指定表单中输入项的一个子集作为绑定输入项(bnding input)，其余的是自由输入项(fee input)。通过给绑定输入项赋以不同的值，可以生成多个表单提交。以SQL查询为例，查询模板可以简洁地表示表单中的所有查询：select*from D where P'，其中P'只包含表单中绑定输入项的选择谓词。绑定输入项的个数即模板的维度。表9-1显示了例9-1表单查询模板的三个实例。

表9-1：查询模板例子，每个模板都有不同的绑定输入项以及相应的表单提交的URL集合和底层Jobs数据库上的SQL查询

绑定输入项	表单提交URL和SQL查询
st	URL: *http://jobs.com/find?src=hp&kw=&st=S&sort=salary&s=go* 查询: select * from Jobs where state = S
kw	URL: *http://jobs.com/find?src=hp&kw=K&st=Any&sort=salary&s=go* 查询: select * from Jobs where desc like '%K%'
st, kw	URL: *http://jobs.com/find?src=hp&kw=K&st=S&sort=salary&s=go* 查询: select * from Jobs where state = S and desc like '%K%'

注意，在实践中，为了生成有效的表单提交，必须给模板中的自由输入项分配值。理想情况下，我们希望不要为模板的SQL查询增加任何额外的选择条件。对于文本输入，我们可以分配空字符串；对于选择菜单，我们可以分配菜单的默认值，期望该值可以用通配符表示。我们注意到，为了减少用户交互操作的麻烦，绝大部分的表单都支持通配符表示。

探寻deep-web站点的问题现在可以划分为两个子问题：

·选择一组合适的查询模板；

·为绑定输入项选择合适的输入值，即采用实际值来初始化查询模板。对于一个选择框，可以使用该选择框的所有选项；但是对于文本输入框，必须预测输入值，而且我们不能假定对于需要考虑的那些值所在的领域存在先验知识。

为了在说明上尽可能简单，我们假设初始化输入项的一组值对于绑定输入项的所有模板都是一样的。但是，实际上有些输入可能是相关的。例如，一个输入项（如cityName)的值可能依赖于选定的另一个输入项（如state)的值，或者多个输入项（如salaryMax和salaryMin)可能会

约束相同的底层属性。

选择输入项组合

我们首先描述如何判断表单中的输入项和输入项组合对于探寻是有用的。现在，我们假定表单中的每个输入项的值都是已知的，也就是说，把文本输入项和选择菜单都统一对待。对文本输入项中的值进行选择的问题将在第10章探讨。

正如所看到的，我们可以把探寻问题建模为选择一组合适的查询模板。在选择一组正确的模板中，从识别我们需要追求的一些目标开始。正如之前所述，简单地枚举所有可能的输入值的笛卡儿积的策略既浪费（给Web爬虫和表单的宿主服务器施加负载负担），又不必要（因为有大量的URL可能不包含数据记录）。

因为我们想尽可能多地发现deep-web内容，很自然地，其处理策略的一个目标是最大化底层数据库的覆盖率（即返回记录的所有数量），同时限制表单提交的总数量。但是，由于我们生成的页面需要添加到搜索引擎的索引中，因此我们也必须解决一些其他的问题。

首先，最佳的策略可能是确定一些能够包含大量结果的表单提交。但是，有大量结果的Web页面并不一定最适合添加到搜索引擎索引中。举个例子，对一个二手车网站，只有一个包含所有待售的Honda汽车的页面是没有用的；相反，如果它有多个页面，包含Honda的每个模型或者不同价格范围则更理想。因此，我们希望生成在每个页面中既不包含太少也不包含太多记录的URL。

其次，虽然一个搜索引擎的主索引很大，它还是远远不够存储从Deep Web抽取的所有页面。由于搜索引擎最重要的目标是根据用户的查询将用户引导到相关的Web站点，我们更希望拥有来自很多站点的丰富且重要的内容。从某种意义上说，生成的URL个数不需要是完整的，但必须足够好以便将流量引导到底层站点。

再次，对于Web站点探寻来说，实际上不需要追求百分百的覆盖率；为一个站点的那些内容丰富的页面建立Web索引就足够了。定期执行的Web爬虫最终会抓到通过探寻得到的页面的超链接（如对于相同的查询，链接到更多的结果，或者链接到相关查询的结果），因此，最终该网站的覆盖率会增长。

总之，为了实现只通过在每个站点提交少量的查询就能够达到较好的（可能是不完整的）覆盖率，我们的目标是为几百万个不同的表单选择查询，而且通过探寻将获取的页面作为搜索引擎索引的好的候选页面。

查询模板的质量

我们可以把以上列出的每个目标作为查询模板的标准。例如，我们绝对不想要一个包含表现层输入项作为绑定输入项的模板。这种模板返回的结果也可以简单地通过一个不包含表现层输入项的模板来提交查询，从而获得返回结果，而且这种处理方式，通常可以生成更少的URL。

包含很多维度的模板通常不是我们想要的，因为它们生成太多的

URL，这些URL中有很多无法返回结果。然而，维度越多的模板能够保证返回越多的记录。

维度较小的模板通常是我们更想要的，因为它们生成较少的URL，但是也有可能这些查询每个都将返回太多的结果。正如之前所述，有太多记录的页面并不适合作为索引。此外，站点可能对在每个页面上显示的实际记录有限制，因此减少了返回的实际记录的数量。

因此，我们期望模板1）不包含任何绑定的表现层输入项；2）不包含维度太多或者太少的URL。从直观上说，模板的维度应该依赖于底层的数据库大小。

现在我们定义一个测试来进一步说明之前所述的标准。在我们描述之前，先举个例子来说明这个想法。如例9-1表单的两个模板，模板T1只有一个绑定输入项st(State)；模板T2只有一个绑定输入项sort(Sort by)，现在考虑这两个模板生成的表单提交集合（如表9-2所示）。

表9-2：通过分析模板的表单提交生成的内容的相似性来确定模板是否为富信息量（informative）

查询模板 （绑定输入项）	模板表单提交 （富信息量的/非富信息量的）
T_1(st)	*http://jobs.com/find?src=hp&kw=&st=Any&sort=salary&s=go* *http://jobs.com/find?src=hp&kw=&st=AK&sort=salary&s=go* *http://jobs.com/find?src=hp&kw=&st=AL&sort=salary&s=go* … 网页内容不同→模板是富信息量的
T_2(sort)	*http://jobs.com/find?src=hp&kw=&st=Any&sort=salary&s=go* *http://jobs.com/find?src=hp&kw=&st=Any&sort=startdate&s=go* … 网页内容相同→模板是非富信息量的

我们注意到模板T1的每个提交都返回不同状态下的job列表。因此，返回的记录不同，从而结果Web页面内容将会很不一样。我们称这种生

成的Web页面内容很不同的模板为富信息量模板。反之，因为没有关键字，T2中的每个提交返回的记录都是相同的（所有job)我们称这种模板为非富信息量模板。从本质上讲，我们的目标是选择富信息量模板，去除非富信息量模板。

因此，我们可以基于从表单提交生成的Web页面的内容区分度(dstinctness)来评价模板。我们通过基于由模板生成的Web页面内容的相似性来对这些页面进行聚类，从而估计该模板生成的不同的Web页面的数量。

如果和表单提交的数量相比，不同的Web页面的数量很小，很可能是如下情况：1）该模板包含表现层输入项，因此多个页面的集合本质上有相同的记录；2）模板维度相对于底层数据库来说太多，因此产生大量的无记录页面，不同页面之间都很相似；3）模板（或者表单）有问题导致出现错误结果页面，这些错误结果页面也很相似。如果模板不存在这种情况，但是还是生成很相似的内容，那可能是搜索引擎索引的边界值，因此不可能对搜索引擎查询有任何影响。

富信息量测试

我们考虑对于给定模板生成的URL，并且下载这些Web页面的内容。我们计算从每个提交得到的Web页面内容的签名，当一个模板生成的Web页面内容计算得到的签名个数远远少于可能的提交个数时，则认为该模板是非富信息量的。

富信息量是根据阈值τ来定义的，该阈值可以通过实验确定。假设Sd

是不同签名的集合，St是所有签名的集合。那么，当|Sd|/|St|＞τ时，表示T是富信息量模板。

相比之下，关于内容签名更具体的细节不是很重要，我们列举了该函数需要的一些重要属性。首先，该签名必须对HTML格式是透明的，因为表现层输入项通常只是改变Web页面的布局。其次，该签名必须和词项的排序无关，因为结果重排是很普通的表现层操作。再次，该签名必须能够忽略页面间的细微区别；因为区别的来源通常是广告，尤其是商业网站，更是如此。这些广告通常在页面边缘处展现。它们增加了页面的文本内容但是并不反映返回结果的内容，因此必须被过滤掉。这些签名不应该包含输入项的值本身。一个二手车搜索网站，如果在邮编为94107的区域没有红色的Honda Civic车出售，则很可能会有这样的错误信息“在94107区域没有红色Honda Civic汽车出售！”(N search results for Red Honda Civic in 94107！）。类似的，对于以{color make model zip}为查询项的结果页面，会有大量的“No search results for{color make model}in{zip}”的结果页。这些页面的唯一区别是搜索词不同，不排除搜索词的签名很可能会把这些页面当做不同的页面，因而认为相应的模板是富信息量的。

在实践中，可能不需要分析一个模板生成的所有提交的内容；而只是需要测试所有可能提交的一个足够大的样本集合就够了。

搜索富信息量查询模板

我们的目标是搜索表单中的富信息量查询模板。我们可以采取一个

非常简单的策略来考虑表单中所有可能的模板，并对每个模板应用富信息量测试。但是，假设一个表单包含n个输入项，存在2^n-1个可能的模板，那么对每个可能的模板进行测试的计算代价很高而且不必要，因此，可以采取增量式的策略而不是遍历所有模板空间，并且只测试那些可能是富信息量的模板。

我们的策略是从单一的绑定输入项模板开始，通过自下而上的方式搜索模板空间。这种策略的基本思路是认为模板的信息量很可能依赖于它扩展的模板，也就是说，它有额外的绑定输入项。如果模板T有维度k，而且其扩展的所有维度小于k（即维度为k-1）的模板都是非富信息量，那么T不可能是富信息量模板。

我们通过所有维度为1的模板开始，测试每个候选模板的信息量。如果有维度为1的模板被认为是富信息量模板，我们就给该模板维度加1，也就是说，构建包含绑定输入项超集的维度为2的模板。

因此，候选模板必须至少包含一个富信息量扩展模板（但该候选模板其本身并不一定是富信息量）。然后测试每个新的候选模板来确定该新模板是否为富信息量。从维度为2的富信息量模板，我们继续以相同的方式构建维度为3的候选模板，如此反复操作。如果某个维度不包含富信息量模板，程序就终止。

注意：当增加一个模板时，考虑到所有的候选输入项。我们可以选择更积极的策略，只考虑富信息量的输入项，即其相应的维度为1的模板是富信息量的。但是，我们相信，在实践上，这种策略会（错误地）

忽略一些富信息量模板。也有很多表单，包含一个主要输入项用于返回结果，而其他的输入项实际上只是作为修饰。例如，一个包含make和color输入项的表单，如果对于make输入项选择默认值，而只选择color这一输入项时，则无法返回结果记录。因此，只有绑定输入项color的表单是非富信息量，而包含make和color的则是富信息量。

一旦搜索终止，我们可以把所有富信息量模板生成的URL增加到搜索引擎的索引中。对于搜索的调整有很多可以完善的地方。例如，我们发现永远都不需要考虑多于三个绑定输入项的模板，或者是生成很多提交的模板。它们不可能是富信息量模板，因此可以很容易地删除这些模板。不需要分析由一个模板生成的所有URL；通常来说，考虑一个足够大的样本就够了。

实验分析表明，我们的算法生成的URL远远少于其他更简单的可选策略。特别地，我们发现生成的URL数量比不采用富信息量测试的最佳的启发式算法少了两个数量级。我们还发现该方法可以高效地为表单确定富信息量模板——在分析过程中，无论测试的模板个数还是下载的模板提交总数在数量上都很小。重要的是，我们发现生成的URL依赖于底层数据库的大小，而不是表单中输入项的个数。

预测输入项值

大量的HTML表单包含文本输入项。此外，有些包含选择菜单的表单必须有文本输入项值才能够返回检索结果。

我们注意到文本输入项通常用于两种方式。第一，通用的输入项实

际上可以接受任何合理的值，在输入项输入的查询词可以返回在后台数据库中包含该词的所有文档。这种情况的通常例子是通过标题或者作者搜索书籍。第二，存在一些键入的输入项(tped input)。这种输入项的值只能是一个定义良好的有限集或者数据类型（如邮政编码），或者属于一些连续的但定义良好的数据类型（如日期或价格）。在固定格式的文本框中输入无效的值通常会导致错误页面，因此识别正确的数据类型很重要。在通用的文本输入框中输入不好的关键字仍然可以返回一些结果，因此挑战在于可以识别关键字的一个有限集合，从该集合中可以抽取很多不同的结果页面。

文本输入项的两种类型可以分别对待。在接下来的部分，我们首先描述一个算法为通用的输入项生成关键字，然后考虑键入的输入项。

通用的文本输入项

在我们描述如何为通用的输入项识别好的候选关键字前，先来考虑（并驳回）一个可能的备选方案。可以想象，我们本可以在各种领域把涉及的查询词列表作为文本输入项值，并把每个文本输入项值和最适合的查询词列表进行匹配。但是，我们很快发现这种处理方式会有太多的概念和领域。而且，对于通用的输入项，即使我们在相同领域符合相同概念的两个分离的表单中识别输入项，相同集合的关键字也不一定在两个站点上都能正常工作。最佳的关键字通常是属于特定站点的(ste-specific)。由于我们的目标是扩展到几百万的表单和多种语言，我们需要一种简单、高效和完全自动化的技术。

我们采用迭代式探测的方法：从高层上，我们指定候选关键字的初始种子集合作为文本框的输入值，构建查询模板，把该文本框作为单一绑定输入框。生成相应的表单提交，下载相应的Web页面内容，从结果文档中抽取额外的关键字。然后，我们用这些抽取的关键字更新文本框的候选值；重复该过程直到无法进一步抽取关键字或者达到了某个候选的停止条件，如有足够多的候选关键字。在终止时，选择候选关键字子集作为文本框的输入值集合。

迭代式探测在过去作为从文本数据库中检索文档的方法而提出(Brbosa 2004，Callan 2001，Ipeirotis 2002，Ntoulas 2005）。但是，这些方法的目标是实现特定站点的最大覆盖。因此，它们采用了和站点相关的(ste-aware)技术，这些方法无法应用于所有的领域。

在高层上，我们定制了如下迭代式的探测方式：

为了确定文本输入框实际上是否是一个通用输入框，在第一次迭代时，使用初始候·选集合在模板上执行富信息量测试。结果表示，通用的文本输入框很可能会被认为是富信息量模板，而其他的输入框则是非富信息量。

·为了选择候选值的种子集合，我们分析了包含该表单的Web页面的内容。我们通过识别与页面内容最相关的单词来选择页面的单词。任何合理的查询词打分方法，如流行的TF-IDF方法(Slton 1983），都可用于选择表单页面的几个最佳查询词。

为了在每次迭代结束时选择新的候选值，我们考虑为模板分析的所

有表单提交页面·上找到的所有选择词集合。排除在很多页面上都包含的选择词，因为这些选择词出现在每个页面上，很可能是HTML样本模板的一部分。我们还排除只出现在一个页面上的选择词，因为这些选择词很可能是没有意义的或者是具有特殊含义，不能代表该表单所表示的站点的内容。

为了对文本输入框选择最终输入值集合，考虑从表单页面或提交页面抽取的所有候·选值，根据这些值检索到最丰富的内容的能力来选择这些值（通过分析从表单提交得到的页面内容）。

我们注意到，为每个文本输入框设置单一的关键字个数的最大值的方式是不合理的，因为表单站点的结果内容的区别可能是几个、几十个到几百万个。我们采用back-off的策略来解决该问题。从每个表单一个小的最大值开始，在这过程中，我们评估了这些生成的URL对搜索引擎流量会带来的影响。如果受到影响的查询个数很高，那么我们就为该表单提高关键字个数限制，然后重启探测进程。

实验分析表明，这里概述的迭代式探测方法对于通用输入框选择输入项值是有效的。相应的表单提交可以展现底层数据库大量的记录个数。有趣的是，我们发现相同表单的文本输入项和选择菜单通常能够展现底层数据的不同部分。我们还可以构建Web爬虫，通过系统生成的URL，能够随时间展现更多的deep-web内容。

键入的文本输入项

实践表明，存在相对较少的类型，如果能够识别出来，可以用于索

引很多领域，因此出现在很多表单中。例如，邮政编码在很多领域被作为输入项，包括店铺定位、二手车、公共记录和房地产。类似的，日期通常在很多领域作为输入项，如事件和文档档案。我们基于两个想法来利用以上的观察。首先，只有键入适当值，键入的文本输入框才可以生成合理的结果页面。通过这个想法，我们使用已知值为流行类型构建富信息量测试。我们考虑有限的和连续的类型。对于有限的类型（如美国的邮政编码和州的缩写），使用已知值的部分样本进行富信息量测试。对于连续的类型，可以根据不同规模次序，测试使用均匀分布值的集合。我们可以使用这样的输入项名称列表，通过手工提供或者随时间进行学习（参见[Doan 2001]），来选择候选输入项，应用于富信息量测试。

[1]即决定提交查询表单时，需要填写哪些输入框，并找到合适的值填写这些输入框。

结论

我们描述了探寻Deep Web内容的方法，使得搜索引擎可以访问这些内容。该系统最重要的要求是它必须是完全自动化的（因此可以扩展到整个Web)，而且可以从任何语言、任何领域来检索内容。有意思的是，这些严格的要求激发我们探索出一个相对简单优雅的解决方案，从而表明简单性通常是解决复杂问题的关键。

今后探寻Deep Web的工作有很多方向。特别地，可以识别表单中存在的特定类型来扩展爬虫的覆盖度。例如，各个域的组合之间通常是相互关联的（如MinPrice和MaxPrice)，输入有效的精心选择的值的组合可以探寻到更多的页面。

参考文献

[1]Barbosa,L. and J.Freire."Siphoning Hidden-Web Data through Keyword-Based Interfaces."SBBD 2004：309-321.

[2]Bergman,M. K."The Deep Web:Surfacing Hidden Value."Journal of Electronic Publishing，2001.

[3]Callan,J. P.and M.E.Connell."Query-based sampling of text databases."ACM Transactions on Information Systems，19（2）：97-130，2001.

[4]Doan,A.，P. Domingos,and A.Y.Halevy."Reconciling Schemas of Disparate Data Sources:A Machine-Learning Approach."SIGMOD Conference 2001：509-520.

[5]"Forms in HTML documents."http：//www. w3.org/TR/html401/interact/forms.html.

[6]He,B.，M. Patel,Z.Zhang,and K.C.-C.Chang."Accessing the Deep Web:A survey."Communications of the ACM，50（5）：95-101，2007.

[7]Ipeirotis,P. G.and L.Gravano."Distributed Search over the Hidden Web:Hierarchical Database Sampling and Selection."VLDB 2002：394-405.

[8]Madhavan,J.，L. Afanasiev,L.Antova,and A.Y.Halevy."Harnessing the Deep Web:Present and Future."CIDR 2009.

[9]Madhavan,J.，S. Jeffery,S.Cohen,X.Dong,D.KO'C.Yu,and

A.Y.Halevy."Web-scale Data Integration:You can only afford to Pay As You Go."CIDR 2007.

[10]Madhavan,J.，D. KO'L.Kot,V.Ganapathy,A.Rasmussen,and A.Y.Halevy."Google's Deep-Web Crawl."PVLDB 1（2）：1241-1252（2008）.

[11]Ntoulas,A.，P. Zerfos,and J.Cho."Downloading textual hidden web content through keyword queries."JCDL 2005：100-109.

[12]Raghavan,S. and H.Garcia-Molina."Crawling the Hidden Web."VLDB 2001：129-138.

[13]Salton,G. and M.J.McGill.Introduction to Modern Information Retrieval.New York:McGraw-Hill，1983.

[14]SpiderMonkey(JavaScript-C）Engine,http：//www. mozilla.org/js/spidermonkey/.

[15]V8 JavaScript Engine,http：//code. google.com/p/v8/.

第10章　构建Radiohead的“House of Cards”

Aaron Koblin和Valdean Klump

这是一个关于获格莱美(Gammy)奖提名的乐队Radiohead[1]的音乐视频“House of Cards”是如何完全根据数据创建出来的故事。在你阅读本章之前，你应该先观看这个视频。该视频权威资料在Gcode项目页面(http：//code.google.com/radiohead)上可以找到。在该网站上你还可以找到一些其他的资源，包括我们用于构建该视频的数据样本，一个允许你通过3D模式查看数据的Flash应用，一些你可以用来创建自己的可视化，以及一个“Making of”的代码视频。当然首先需要找出这些代码。

这一切是如何开始的

2007年9月，我收到一封来自James Frost的电子邮件，询问我是否对基于数据来创造一个音乐视频感兴趣。James是一个天才视频导演，他为Coldplay、Norah Jones、Pearl Jam以及很多其他流行的艺术家做了很多工作。他看过我的“飞行模式”(Fight Patterns)项目（见图10-1）——使用空中交通的GPS数据对商业飞行模式和密度进行可视化——而且希望碰面并讨论制作一个可视化音乐视频。

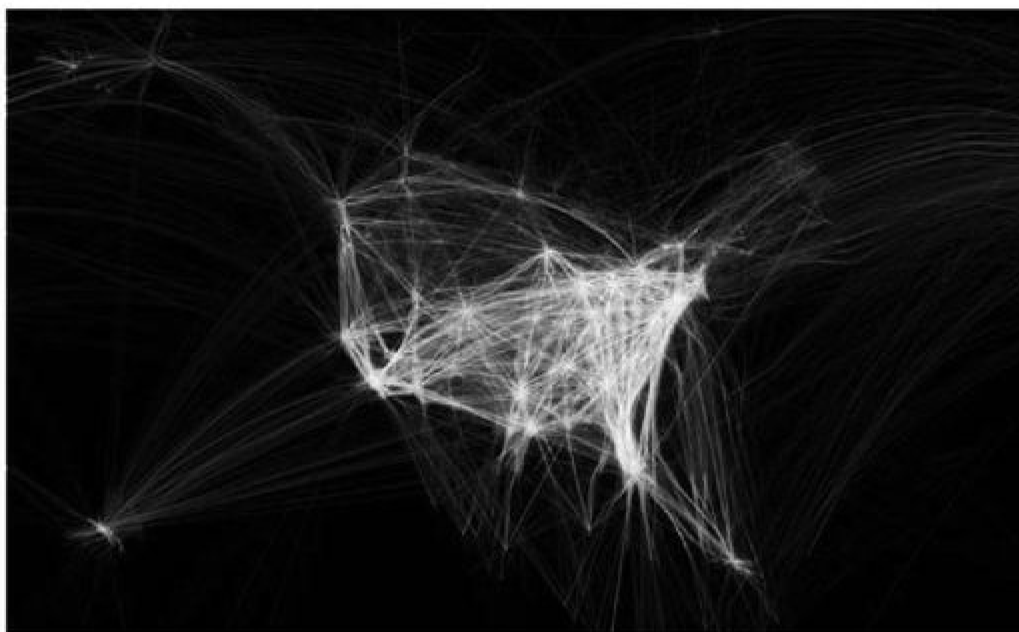


图　10-1：来自“飞行模式”的静态图片（2005）（见彩图22）

两个月后，James、他的制片人Justin Glorieux和我在洛杉矶的咖啡厅会面，我们讨论了一些想法。我向他们展示了自己做过的一些项目以及我认为有助于可视化制作的一些技术。我们讨论了使用Proccessing编程语言的一些可能性，该语言已经广泛应用于数据可视化。最终通过这种

方式为Interpol乐队制作了“Rest My Chemistry”视频，该视频在2008年3月发行。如果你从来没有用过Processing语言，我强烈建议你访问它的网站：http：//processing.org/，并找出相关代码。就我所知，Processing对于艺术家、设计人员或者任何对动态数据可视化感兴趣的人而言，都是最佳的编程语言。

我们讨论的另一种可能是对激光传感器收集的数据进行可视化。我在加州大学洛杉矶分校(ULA)的嵌入式网络传感(CNS)中心做一个项目时，第一次接触了这种技术。CENS使用激光来检测光线是如何穿透森林的覆盖的，我被渲染后的图片所蕴含的内在美深深地震撼了。James看完一些例子后，同意了我的建议，而且使用激光来制作影片的想法给他留下了深刻的印象。他问：“你的意思是不用摄像机来拍摄视频？不用视频就可以拍摄视频？”他马上意识到可以做一件从来没有做过的事情。在那之后不久，他以这种想法开始制作Radiohead的音乐视频。

希望你能够了解该影片是如何制作的并从中找到自己的工作灵感。在这一章，我将首先讨论我们捕捉数据所使用的工具。然后，我将谈到数据本身即视频拍摄，以及后期的数据处理。最后，我们一起来看我在GCode网站上提供的数据可视化代码，并讨论你可以自己如何发挥它。

[1]Radiohead是一支来自牛津大学的乐队，通常中文翻译为“电台司令”，实际上是指收音机粉丝（因此本书对该词不做翻译，保留原文），乐队由Thom Yorke（主唱，吉他手）、Ed O'Brien（主唱，吉他手）、Jonny Greenwood（吉他手）、Colin Greenwood（贝司）和Phil

Selway（鼓手）组成，是20世纪90年代初受华丽壮美的竞技/舞台摇滚所影响的另类乐队。

数据捕捉设备

如果没有一些精密复杂的数据捕捉设备，“House of Cards”视频就不可能成功。当你观看视频，你会注意到有很多场景，从静态的郊区风景到歌手Thom Yorke所唱的动态点云(pint clouds)场景。为了同时捕捉对Thom的歌唱和风景相结合的特写，我们需要使用两种不同的设备：Velodyne公司的Lidar（激光雷达）和Geometric Informatics公司的可视化系统。

Velodyne公司的Lidar

Velodyne是坐落在美国加州的San Jose南部的一家公司，由两个人来运营，他们业余时间会参加机器人大赛如Battle Bots和Robot Wars竞赛。该公司生产扬声器、立体音响设备和强大的激光扫描设备，包括我们用于捕捉“House of Cards”视频所需要用到的风景和派对场景的HDL-64E Lidar（激光雷达设备）。HDL-64E真正声名显赫的原因是它被成功地应用于2007年DARPA城市挑战赛(Uban Challenge)的一些交通工具中，包括获得冠军的团队，成功地把它应用于环境和地形来提高视觉力。在某些情况下，它是这些交通工具的唯一视觉系统。

Velodyne公司的HDL-64E Lidar是一个扫描仪，包含64个激光发射器和64个激光检测器。它围绕一个圆圈旋转，以360°水平方向和26.8°垂直方向旋转收集数据，每秒可以扫描超过100万个数据点，收集到接近5MB的原始数据。默认情况下，Lidar以600RPM（10Hz)的速率旋转，

可以通过系统计算机的序列化端口发送文本命令来调整速率，速率范围介于300～900RPM。在扫描静态风景时，为了达到最佳分辨率，我们采用了其最高设置900RPM。

Lidar的可达范围随环境的反射率而变化。例如，沥青路面可以拍摄50米以内的范围，而汽车和树叶（它们的反射率更高）可以达到120米。Lidar最小的可达范围是3英尺，如果更近，那么光线反射回检测器太快，以致无法计量。

发射-检测设备被划分成两组，每组包含32个激光斜岸(Lser bank)，如图10-2所示。上斜岸指向斜上方，与仰角的上半角方向一致；换句话说，它扫描了Lidar的上半个垂直视图域。与此相反，下斜岸扫描了仰角的下半个视角。通常上斜岸仰角更高，因此对于处在远处的一个物体，上斜岸的激光脉冲可以到达的位置范围比下斜岸的要大。为了在更长的距离获取好的分辨率，上斜岸的激光被触发的次数是下斜岸的被触发次数的三倍。从视频的静态图片中可以看到这种方式对我们的数据所产生的影响，如图10-3所示。

Lidar通过发射一束光（即激光束）并测量该光束返回的光量来获取一个数据点。随着设备的运行，这64个发射器每个都释放持续5ns的光脉冲；然后使用透镜对脉冲进行聚焦，再用平面镜把光线反射到外界环境中。如果该光线在外界环境中照射到某些物体，会有一小部分光线反射回Lidar。这种反射回的光线会穿过激光接收透镜和太阳光紫外线过滤器。太阳光紫外线过滤器能够限制太阳所带来的光量；如果没有这个过

滤器，自然太阳光线会降低系统的灵敏度，从而给数据带来很多噪音数据。

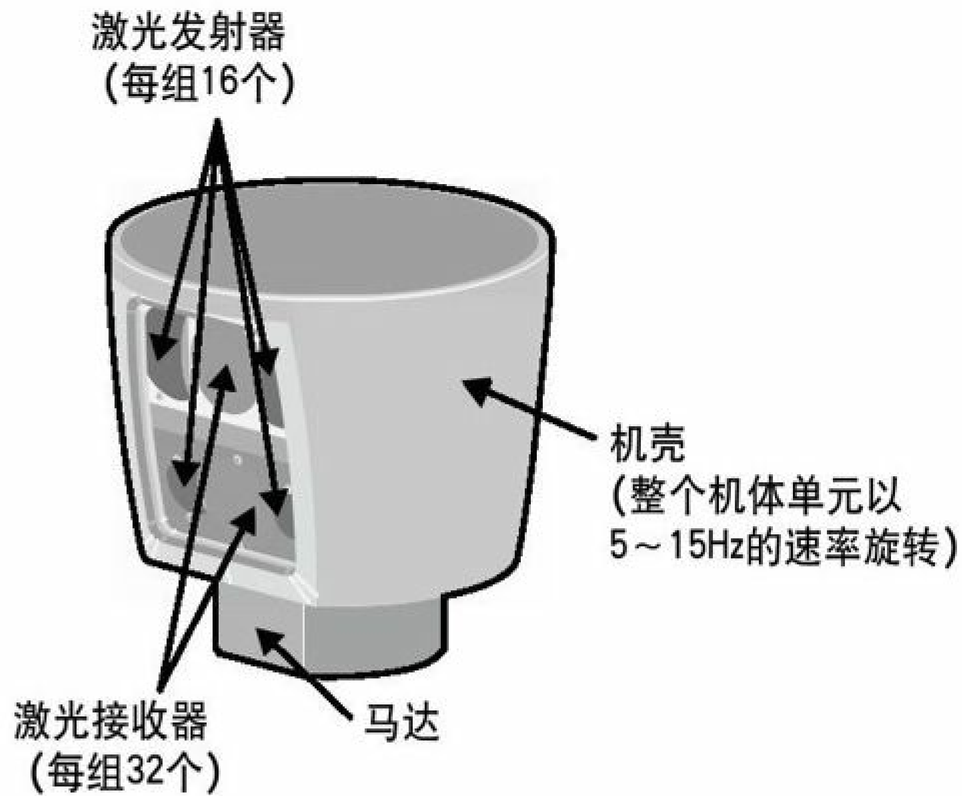


图　10-2：Velodyne公司的Lidar（图像来源于Velodyne公司）

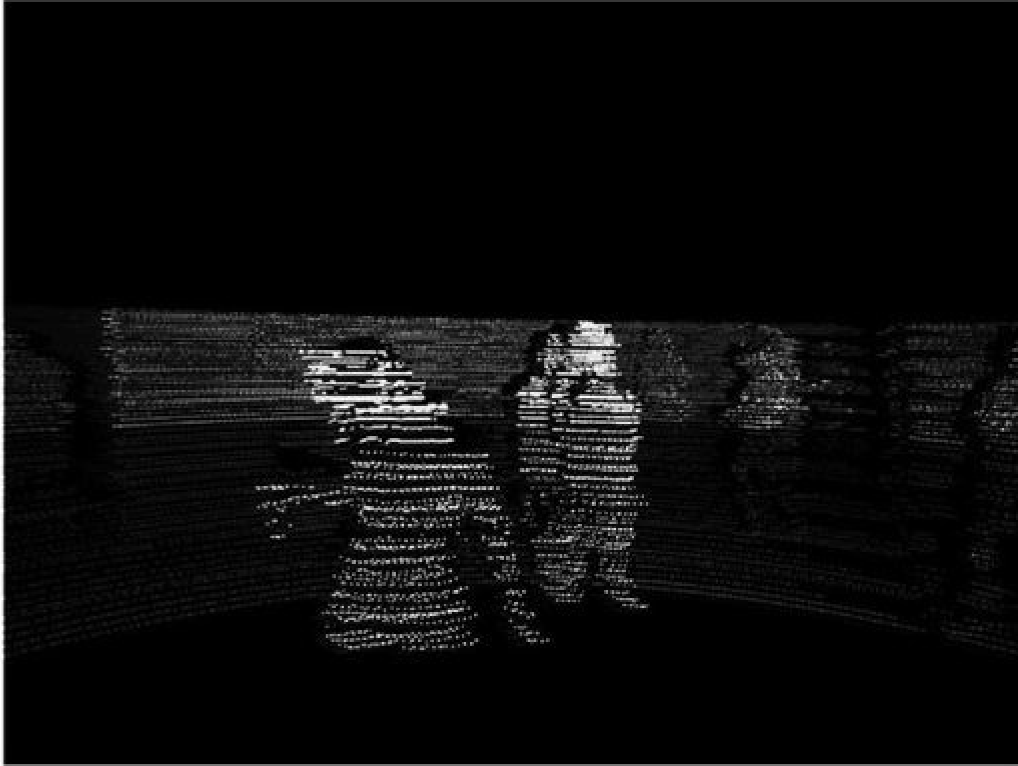


图　10-3：派对场景的一张静态图片，通过Velodyne的Lidar拍摄。注意该图片上方的分辨率更高，这是由于在上斜岸的激光的触发速率更快（见彩图23）

发射回来的光线经过太阳光过滤器后，接收透镜会把返回的光线聚焦于名为Avalanche Photodiode(APD)的光检测装置上，该装置生成和接收的光线强度相当的输出信号。将APD的输出信号进行放大并把模拟信号转换成数字信号。然后把数据发送给数字信号处理器，该处理器决定返回信号的时间。脉冲的强度和返回时间构成了一个数据点。正如我之前所说，HDL-64E模型每秒创建了超过100万的数据点，即每秒超过5MB的原始数据。

等数据创建完成后，传感器通过标准的100BaseT以太网端口把结果

数据输出给用户。数据以等于旋转速率的帧速率（600RPM=10Hz帧速率）不断地从以太网端口输出。以太网数据包中记录了每个发射器-检测器的距离、密度和角度相关的数据。然后通过以太网包捕获程序捕获这些数据，而在我们这个项目中，将会把捕获的数据保存到硬盘上。

Geometric Informatics公司

Lidar是对户外景色进行可视化的神奇工具。它可以根据距离检测一个镍币的大小，在较大的空间中工作良好，但是对于人脸的轮廓和细节的识别，它还不够完善。为了对Thom Yorke的歌唱进行特写，我们需要一些其他的东西。我们需要有更好的视觉。

当谈到如何近距离拍摄特写时，我突然想起在2005年SIGGRAPH会议上发现的叫做Geometric Informatics公司的设备。（题外话：如果你有任何很酷的数据可视化技术，请到每个会议上展示这些技术，我有可能会参加这些会议。谢谢。）Geometric Informatics公司在该会议上有个展台，演示其称为GeoVideo的系统。

GeoVideo是一个实时动作捕捉系统，它尤其适合捕捉人脸的几何图形。在近距离拍摄时，它的效果明显优于Lidar系统，它能够检测到0.2mm的数据点，而Lidar只能检测到2cm的数据点。有了GeoVideo，我们能够捕捉到Thom Yorke唱歌时的具体细节。你在视频开始看到的点云数据(pint cloud data)就是通过GeoVideo捕捉的。

如果你认为图10-4所绘的图看起来挺简单，那是因为该设备外表看起来确实没有很多特别的地方。系统看起来像一个米色的箱子，每面有

个胶卷，有两个透镜。一个透镜把一个区域的光线投影到箱子前面，而另一个透镜则捕捉该数据。光线区域包含60万个直角的网格，该网格在效果上形成一个即时等高线图，投影到传感器的前面。然后，传感器读取每个直角点作为一个数据点，然后把这些原始数据以每秒54MB的速率，不加任何改动地输出到一台计算机上。传感器每秒可以捕捉180帧。

采用GeoVideo的方法把光线区域投影到物体上的优点是，无论传感器前面需要被投影的是什么东西，都不需要在其上画上一些网格，不需要穿着用于动作捕捉的“衣服”，也不需要坐在一个包含参照物标记的绿色屏幕前。光线投影创造了一个即时、可移植的参照图，其使用之简单达到了令人难以置信的程度。

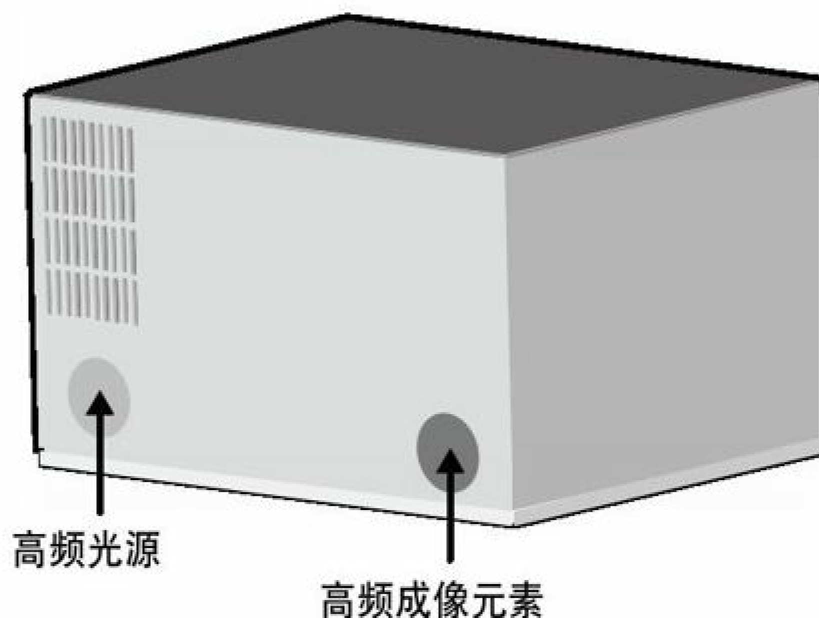


图 10-4：Geometric Informatics系统（图像来源：Geometric Informatics

公司，见彩图24）

GeoVideo系统也支持纹理映射，这意味着它不仅可以捕获数据点，而且可以捕捉这些数据点之间的纹理。综合而言，它可以达到对物体或者人脸进行精确的3D表现的效果。但是对于“House of Cards”视频，我们决定放弃纹理捕捉，而只使用其数据点。即使如此，我们还是进行了大量采样。其结果是你在视频开场看到的Thom Yorke的数字化的“点云”。该图像看起来和Thom Yorke并非完全一致，而是像个数字化的头像或者精灵——至少，我觉得是这样。看了两个版本的数据——包含纹理的和不包含纹理的——我认为不使用纹理的那个版本看起来更有趣。有了纹理，他看起来有点像视频游戏中的一个角色。有时抛弃一些数据会使可视化效果更美丽。

两种数据捕捉系统的优点

值得一提的是，自从为这部影片工作后，我开始对在同一个项目上使用多个数据捕捉系统变得很感兴趣。混合技术在创造力上可以产生事半功倍的效果。我们本来可以只使用Lidar系统或者GeoVideo系统来拍摄整部影片，但是我很高兴我们采取了不同的方式。这可能使项目更复杂，但是它使我们变得更具有创造性。对于我们需要捕捉的东西，它也给我们带来了更多的灵活性。

在寻找设备方面，你也可以从我的经验中获取一些教训。第一，如果要为数据可视化寻找设备，那么要找遍每个地方。有很多令人兴奋的传感技术被开发出来但却从未使用过。如果你准备启动一个可视化项目，首先在贸易展览上或者你所在地的大学做一些在线调研。确认是否有新的、你未曾考虑过的方式可以捕捉数据。不同的设备可能会给你的工作增加新的主题或者显示出你之前没有看到的数据。总之，尽力去寻找可以给人震撼的可视化技术。

第二个结论是一个警告：如果你只使用一种设备，那么你的工作看起来可能仅仅是该设备的演示。如果我们只使用Lidar来创造“House of Cards”视频，我有点怀疑该视频是不是会成为“Lidar视频”。通过同时使用GeoVideo系统和Lidar系统，最终产品不会被贴上某个产品的标签。单一工具是无法定义该作品的。混合使用两种数据捕获系统使得该视频后面的故事变得更有趣。

数据

在我讨论拍摄之前，我想先展示一些样本数据。这些数据其实非常简单。以下是我从Thom Yorke的歌唱数据中任意抽取的三个数据点。它们在文件2067.csv中，它是该视频的第1067帧。因为每秒包含30帧，而且第一个帧是1001.csv，这些数据点可以在视频播放约36秒时看到。你可以在GCode网站上看到这个数据文件，以及一些其他的帧：

70.05，162.48，-79.32，122

70.23，165.26，-78.82，112

70.46，168.00，-77.55，95

这些数据以（x,y,z）的强度格式表示。我们捕捉到的所有数据最终都转化为这种格式。

（x,y,z）值是相对测距。GeoVideo系统和Lidar系统相似，包含（0，0，0）点，基于该点构建所有其他点。因此，70.46是指该点在x轴距离0点是70.46个单元距离。你可以任意扩展这些点。其强度范围是从0（0%白色）到256（100%白色）。

在GCode网站，你可以找到Thom歌唱的2000帧有效数据文件，它们组成刚刚超过1分钟的视频。也可以访问音频数据。我们还包含了两个静态景观的有效数据文件：城市及其尽头的小路。它们在可视程序所在的压缩文件HoC_DataApplications_v1.0.zip中。

捕捉数据，即“拍摄”

2008年5月，我们在佛罗里达州的Palm Beach县花了一个周末还多的时间采集“House of Cards”视频的点数据。Radiohead那时正在那里巡演，而且他们为了减少巡演时的旅行，想要拍摄音乐视频。想要了解该产品的幕后花絮，请在GCode网站检索本章最开始处提到的“Making Of”视频。

户外Lidar拍摄

我们到达佛罗里达州之后所做的第一件事就是在制片小组租的一辆旧面包车的后面安装上Lidar。我们使用该面包车来捕捉在视频中看到的静态的景观数据，如城市及其尽头的小路。

不像DARPA城市挑战赛的汽车，我们没有把Lidar放到汽车的上方，而是把它倾斜了90°度，然后装到面包车的后面。这意味着激光可以垂直扫描环境。如果对这种拍摄方式感到困惑，你可以想象一个灯塔倾斜在一边，粘在车的后面，像一个管道的尾巴。这意味着激光器从街道旋转扫描到天空，然后再回来。每分钟这样反复旋转900次。

我们这么做是因为它能够提供非常高分辨率的区域扫描。而且事实上，在后期处理中，我们从64条激光中只分离出一条激光，这是因为它们都非常有效地扫描了同样的东西。随着面包车向前行驶，激光每次旋转都扫描了环境的唯一独特部分。

图10-5的景观就是通过这种技术捕捉的。面包车径直行驶在街道的

中央。你是否发现街道上的线和街道本身垂直？这是因为Lidar是挂在背面，而且是朝下的。你可能还会注意到在公寓塔楼边上有曲线，这是由于面包车的行驶和Lidar的旋转角度交叠在一起。

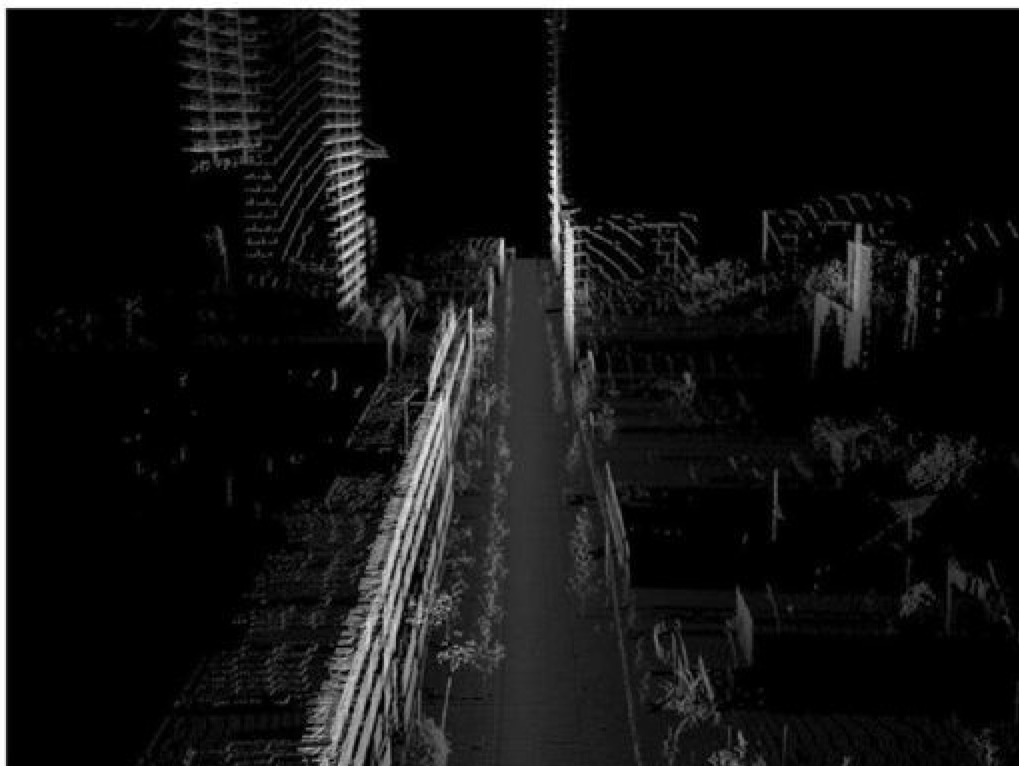


图　10-5：Velodyne Lidar的一个激光器捕捉到的数据（见彩图25）

拍摄过程进行得非常顺利。制片组已经找好了位置，因此我们径直开到每个场景，并顺序扫描它们。当我们到达了想要扫描的地方，我们就会放慢速度到10mph左右，司机会尽量把车开得平稳，然后我们就开始记录。

和摄像机不同，Lidar不会启动和停止。相反，它总是运转着，不断地旋转和输出数据。因此，我们不需要打开或关闭它，而只需要知道什么时候开始记录数据。当要开始记录数据时，我们的导演助理Larry

Zience会以对Velodyne的现场工程师Rick Yoder喊“开动电脑”作为指示，Rick Yoder就会开始收集数据点。（这对于一些制片组成员而言，显得有些可笑，因为通常导演会说“开动摄像机”。）然后，Rick点击他的笔记本电脑上的一个按键，Lidar数据就开始输出到他的硬盘上。当Larry喊“停”时，我们就停止记录。

后来，Rick发给我一封信，是关于和制片组一起工作的感受：

我内心的一个感受就是这打破了一位摄影师要找到合适的镜头角度是非常困难的习惯看法。传统的摄像导演不习惯于从一堆扫描中构建模型，然后在后期处理时操纵相机的角度。他们会说：“我们想要把相机从这里移到那里，然后向上移动大约20英尺，而镜头保持约45°。”我们会说：“好极了。我们只需要把扫描器放在场景中间，你就可以在后期处理中这么做。”

图10-6显示了另一处景观。注意电线为什么会是弯弯曲曲？这是一个很简单的原因：面包车在不平滑的路上上下颠簸产生的自然的曲线。通常，这些“错误”可以通过陀螺仪、加速器以及其他精密设备来避免。在我们这种情况下，我们想要这样的“错误”。不但处理起来代价更低、操作更简单，而且它使得数据也更有趣（至少我这样认为）。完美是值得赞美的目标，但是它并非总是最具有创造性。



图　10-6：Velodyne Lidar捕捉到的另一幅景观图片（见彩图26）

室内Lidar拍摄

我们还把Lidar用于室内拍摄影片背景。它被用于捕捉视频中3：30和3：55时刻的派对场景。和景观场景不同，我们在这部分拍摄中使用了所有的64个Lidar激光器的数据，而不是仅仅一个。这是由于派对场景是动态的——Lidar每次旋转，点数据都会随之变化——这意味着这些点数据在视频的每个帧都有变化。因此，你看见场景中的人们正在运动。在这部分拍摄中，我们使用了正常水平放置的Lidar，这是数据以水平线方式显示的原因。

为了创建派对场景，我们从邻近的学校招募了一些电影专业的学生。有些学生精心打扮了一番，认为他们会出现在Radiohead视频中，

这是他们展现自己的机会；他们几乎没有意识到，事实上我们真正需要的只是他们的身体形态。抱歉，伙计们！

如果你数一下图10-7所示图片的水平线，你会发现有64条。注意为什么图片的上半部分更亮？那是由于Lidar上方的32个激光器比下方的触发得更快。正如我之前所述，Lidar这种构建方式是由于它通常扫描很大的地形空间，要到达水平升起的地方需要更高的分辨率。

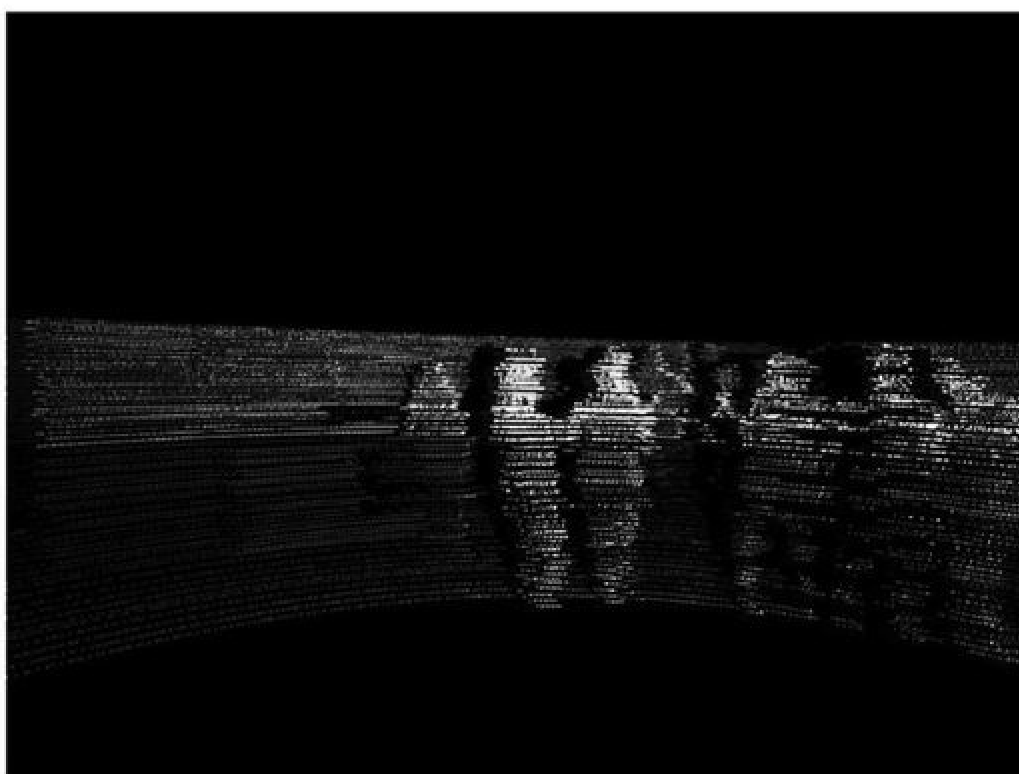


图 10-7：派对场景的一个静态图片，Velodyne Lidar捕捉，数据有64条线，每条线由Lidar的一个激光器生成（见彩图27）

遗憾的是，Lidar的分辨率很低，每个数据点约有2cm，这也是图片看起来很模糊的原因。在我看来，这一点反倒给视频增添了一丝韵味。派对通常是由那些不太认识的人组成的，而这种可视化方式能反映出这

种隔离的“味道”。然而，Lidar的低分辨率无法满足对Thom Yorke拍特写。因此，我们使用GeoVideo系统。

室内GeoVideo拍摄

Thom Yorke的点云图、他的“爱人”（由Lauren Maher扮演，在视频的1：05时刻你第一次看到她）以及一些其他场景都是通过Geometric Informatics公司的GeoVideo系统采集的。

GeoVideo系统能够实现令人震撼的现实主义。如果你看了Geometric Informatics公司的网站上的演示视频，你会发现其图片的质量比我们视频中的点云要高得多。它的可视化也没有我们视频中遇到的干扰和错误。

我们的视频质量低的原因是我们故意这么做。导演James Frost并不想要一个完美的Thom Yorke的视觉化身，他想要看到的是Thom Yorke深情、缥缈的视觉幻影（见图10-8）。当观看视频的开场时，在我眼里，他不是Thom Yorker这个人，而是这位歌手的灵魂。我们在机器里看见的是幻影。

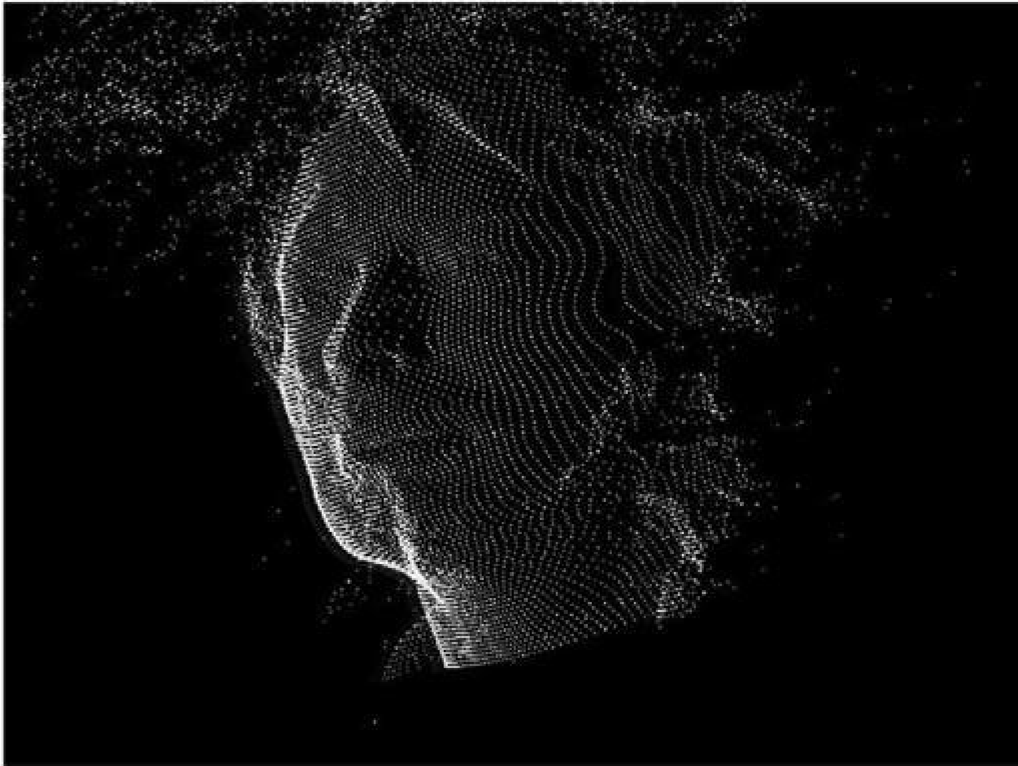


图 10-8：歌手Thom Yorke在Radiohead视频中的一张静态图片（见彩图28）

可视化中数据质量的低下和频繁错误，也使得获取数据看起来很困难。这种似是而非的困难增强了故事性。一张清晰的图片无法表达我们想要表达的意思。

在后期数据处理中，我们没有对数据中的干扰进行干预；相反地，我们为数据集创建干扰。制片公司购买了很多道具来打破现场的数据，包括把一些很小的镜子粘到一面有机玻璃上，在扫描器前洒一些羽毛，在Thom前把水倒在有机玻璃上来制造“流水”景象。最后，用镜子把数据干扰成无组织的方式效果最佳；羽毛在数据干扰上的效果差强人意；而水吸收光线，在数据集中只是生成一些空白点。

室内拍摄花了大约10个小时的时间，包含GeoVideo和Lidar。对于所有包含Thom的场景，为了防止数据丢失，我们很小心地把数据存储到了多个硬盘上，因为担心这些数据万一丢失，我们可能就无法再拍摄出同样的效果了。

处理数据

所有的数据都捕捉完成后，开始了数据处理工作。我们做的第一件事是把原始的Lidar数据发送到510 Systems公司，一家在加州伯克利的工程公司，对处理这种类型的数据有很多经验。该公司把该项目分配给其内部的Lidar数据大师Pierre-Yves Droz。他在把Lidar数据转换为可使用格式领域是专家。

我们把数据以DVD方式邮寄给Pierre之后，他为我们做了两件事情。

首先，对于景观场景，他把一个激光器和其他63个激光器分离开，为这个激光器的点创建了一个数据集。其次，他把所有的原始Lidar数据，包括动态派对场景数据，都转换为包含（x,y,z）和强度的很多单点数据。

其次，为了对原始数据进行转换，Pierre需要知道每组Lidar激光发射器和检测器的精确位置和方向。计算信息是由Velodyne公司提供的，对于每个Lidar单元，其参数都不一样。Pierre还使用面包车的速度来帮助计算Lidar在真实世界中旋转时运动有多快。

最后，我们提供给510 Systems公司大约4GB的原始数据，却生成了接近50GB处理后的.obj格式的文本数据。

后期数据处理

我们一处理完数据，就把它交给The Syndicate公司，在加州Santa Monica的一个视觉效果工作室。它把场景渲染成二维，并添加了粒子流效果，在视频开始约1分钟左右，你可以看到该效果。

The Syndicate工作室的粒子专家Brandon Davis致力于这个项目。他发给我一封邮件，告诉我为什么这个项目很不寻常：

一开始，从可视化角度考虑，Radiohead项目有很多不寻常的可能性。有了生动的数据集合，你就会充满奇怪的“矛盾”：依赖于视觉的数据在视觉上却可以看做独立的。这真的是“第二景象”，能够带一个人欣赏其看到的，并从不同角度来享受这场视觉盛宴，暴露景象的差距。这打开了一些真正独特的图像大门。

Brandon Davis继续描述他是如何处理“蒸发”效果，你将会在视频中注意到这些效果：

客户想要随着时间而退化的数据集，似乎这些点正在被一场虚拟的风刮走。从刚开始，我们看的就是两种显著不同的数据类型——环境的静态Lidar点云和歌手Thom Yorke的动态生动的点云。我们认为后者操纵起来会更有挑战性。静态的数据集相对更容易操纵，因为所有你需要做的是随着时间取代这些点，因此我们知道选择性地触发受重力场影响的部分数据集不会太困难，从而创造出一种建筑物和树都随着喧嚣的巨浪一起蒸发的效果。正如我们所预测的，在静态数据集上生成该效果只

是把跳跃的边框移动到点云，在一帧帧的基础上完成剪辑测试。如果一个点在某个特定帧的内部，就把该点传给粒子系统，这样它可以自由地通过噪音区域，否则它就保持静止状态。这是非常有交互性的，容易控制，且基本上迭代迅速。

然而，Thom歌唱的动态点云却是另一回事。Brandon继续说：

从我的角度看，动态点云是该项目的最大挑战。虽然你可以使用和静态数据集一样的方法，但是动态数据集有其自身特有的问题：它会刷新每个帧！只要你发布一个点，并把它传给一个粒子系统，原始的源点就会反弹到新的位置，基本上重新生成每个帧。因此，虽然我可以指定数据点在他的鼻子上落下，然后再飞走，随后的每个帧都有一个新的点来取代飞走的那个点。

最后，The Syndicate找到了一个简单的方法，使用2D面具增加动态点云的衰减效果：它在3D点云上给面具添加了一层吹走的粒子。这意味着不存在像静态景观那样完美的一对一的粒子衰变，但是我认为这些变化是无法察觉的。

当我咨询导演James为什么他在视频初始时增加了粒子衰变的效果，他说：

“House of Cards”本质上是一首爱情歌曲，和爱情或友情一样，在那里有关系破裂。它们不仅仅停止，而是有一系列的事件活动，一种催化剂导致它们终止或衰落。对于视频，我想在更大范围内探索这个主题。在我看来，在生活中我们对于周围的世界知之甚少。我的想法是基础设

施可以在我们周围大规模地崩溃，但是由于我们没有感受到其影响，我们作为人类已经达到对什么都满不在乎的地步。但是，最终该事件会产生连锁效应，而且最终会在我们个人层面上带来影响。金融机构正在发生的事就是一个最佳的例子；一家银行倒闭了，但是除非你把钱存在那里，否则你就不会去关心它。然后你的钱所在的银行也倒闭了。

当完成了所有的后期处理，编辑委员Nicholas Wayman Harris和我们一起编辑了剪辑。视频终于完成了。

发布视频

“House of Cards”视频是在G公司网站上首映的第一个音乐视频，它在2008年7月11日发布。G公司站点包含了一些该视频的数据，因此你可以创建自己的可视化以及3D数据可视化工具。G公司的创新实验室开发了这个站点。

该可视化工具是由我和朋友Aaron Meyers一起用Flash开发的。它允许观看者在播放视频期间实时旋转云点数据。对我来说，这正是数据美丽之处。Flash应用允许你实时从任意视角查看视频，这是传统的视频记录所不能及的。你可能旋转Thom Yorke的脸庞，这样他从远处望着你，有效地把持住他的脸作为你的面具，你就可以透过他的眼睛。在我看来，这种效果非常强大。它使得音乐视频在某种方式上变得有形，我怀疑很多人都没有过这种体验。

我们还发布了一些数据（把数据开源）以及通过Processing编程语言开发的视频创建工具。然后，我们鼓励人们下载数据来创建他们自己的视频。

我想为视频创建工具分享源代码，以便展示通过Processing创建你自己的视频时多么简单。以下是输出Thom Yorke歌唱的帧的代码：

```
   import processing.opengl.*;
   int frameCounter=1;   //Declare a variable to store which frame
we're dealing with
   void setup（）{  //Here we set up the program
   size（1024，768，OPENGL）;   //This is the render size.We'll use
```

```
OpenGL to draw as
   //fast as possible
   //frameRate (30) ;  //Uncomment to watch the animation at 30 frames
per second.
   strokeWeight (1) ;  //Draw lines at a width of 1, for now.
   }
   void draw () {  //Here we state the things we're going to do every
frame
   background (0) ;  //We'll use a black background
   translate(width/2, height/2) ;  //The data has 0, 0, 0 at the
center and we want to
   //draw that point at the center of our screen
   translate (-150, -150) ;  //Let's adjust our center slightly
   scale (2) ; //Let's draw things bigger
   //rotateY(frameCounter/50.0f) ;  //If uncommented,this makes the
data rotate over
   //time
   //rotateY(mouseX/150.0) ;  //If uncommented,this uses the mouse's
horizontal
   //location to adjust the rotation
   String[]raw=loadStrings(frameCounter+".csv") ; //Here we load the
current frame
   //data into an array
   for(int i=0; i<raw.length; i++) {  //Now we loop through each line
of the
   //raw data
   String[]thisLine=split(raw[i], ', ') ;  //For each line we're going
separate
   //each parameter
   float x=float(thisLine[0]) ;  //Now we make a decimal variable for
each
   //parameter
   float y=float(thisLine[1]) ;
   float z=float(thisLine[2]) ;
   int intensity=int(thisLine[3]) ;
   stroke(intensity*1.1, intensity*1.6, 200, 255) ; //We set the color
of each point to
   //correspond to the data's
   //intensity value
   line(x,y,z,x+1, y+1, z+1) ;  //Here we draw a little line for each
point; this
   //is much faster than a more complex object and//we'll be drawing
a lot of them
   }
   frameCounter++; //Add one to the frame variable to keep track of
what frame we're
   //currently on
   if(frameCounter>2101) {  //If we get to the end of the data we'll
```

```
exit the
   //program
   exit（）；
   println（"done"）；
   }
   //saveFrame（"renderedFrames/"+frameCounter+".tga"）；  //This would
be a way to save out
   //a frame
   //*remember you're saving files to your harddrive! *
   }
```

它没有数据那么美（毕竟本书不是《代码之美》），但是它的效果很好。正如代码所示，它允许你从头到尾观看Thom Yorke的歌唱，但是有一些修改，你可以定制这个过程。以下是两个修改实例，基于之前的代码做出注释。第一个修改实例是：

```
   rotateY(frameCounter/50.0f）；
```

把绘图函数的起始部分的这行代码取消注释，将会导致Thom的脸随着帧的增加而向y轴旋转。

第二个修改实例是：

```
   rotateY(mouseX/150.0）；
```

把绘图函数起始部分的这行代码取消注释，允许你对鼠标函数进行旋转。你现在可以随着帧的输出移动Thom的脸。

我相信你能够想到一些其他方面来修改。很多人做了我从来没有想到的事情，这正是我期望的。对所有的帧进行渲染（把最后一行取消注释）并放在一起生成一个视频，就可以制作类似QuickTime Pro、Final Cut或After Effects这样的节目了。其他人创建的一些视频令人印象深

刻。在Youtube的“House of Cards”组
(htp：//www.youtube.com/group/houseofcards)上看到这些视频。

这一切真的都非常简单；只需要在开始实践时有一些美丽的数据。

结束语

在写本章时，我有7个想法，在你通过美丽的方式来捕获和表示自己的数据时可能会觉得有用：

1.通过新的方式看普通的东西会使它变得很特别。

你不需要扫描月亮、热带岛屿或者一位流行模特来获取美丽的数据。通过新的方式来观察普通的东西可以产生相同的效应。对于“House of Cards”视频来说，我们扫描了一个人的脸以及一些郊区建筑。通过新的方式来观察这些平凡的东西，以及通过新的可视化技术，我们可以使事情变得有趣。

2.讲一个故事。

显然，如果你有一首令人震撼的歌，它对你的工作会有帮助；但是，也尽可能地根据你的数据讲述故事。给这些数据增加一些含义要比仅仅展示数据更加生动。

3.使用多种可视化技术比单一技术会更有趣。

正如我之前所述，“House of Cards”视频是通过多种技术如Velodyne Lidar和GeoVideo系统，以及粒子衰变后期处理效果而变得更加强大。如果我们所做的所有都是可视化原始Lidar数据，那么它就不会很有趣。

4.想想数据，而不是真实世界。

当我们给有机玻璃板增加一些镜面，在Thom唱歌时把它移到他脸前时，我们没有想在视频相机里它会是什么样子。数据即是产品。当你看

一些你想可视化的东西时，想想你可以从中获取的数据。

换句话说，尝试使一件很离奇的事情变成现实，然后感受这种现实。这会使你的故事古怪离奇。音乐视频通常描述离奇的一面。问问自己，你该如何操作数据使得它更稀奇、有趣和与众不同?

5.不需要使用全部数据。

我们使用GeoVideo系统来扫描Thom Yorke生成的数据超出我们想要的。通过大范围抽样，我们生成了更有趣的数据点。我们想要的不是一张照片。

6.把数据开源。

通过使得其他人可以拥有这些数据并创建他们自己的“House of Cards”视频版本，使得每个人可以创建自己认为最美丽的视频。每个人总是对于什么是最好的有自己的观点。让人们“纵容”自己的观点。如果你见到他们的作品，你会感到高兴且满足，你可能永远都不知道，有些人做了一些意想不到的事情。这是件好事。已经有超过10万人从GCode网站下载了数据并创建了一些伟大的视频。

7.和Radiohead乐队一起工作。

可能显得有点不够严肃，但是毫无疑问，我们很幸运能够和世界上最有创造性的乐队一起工作。而且不仅仅是他们，而是整个团队。只有和一些无比卓越的天才一起合作才使得这个视频变成可能，包括James Frost，以及Velodyne、Geometric Informatics、510 Systems和The Syndicate的员工。和这些比你更具智慧的人合作，你的项目会受益良

多。

第11章　都市数据可视化

Michal Migurski

引言

数据因何而美丽，它又来自哪里

美丽的数据有趣、有用、公开且免费。数据必须使它的收集者、观众或者某个机构能够对它感兴趣。它必须对那些感兴趣收集和维护这些数据的人有用，可以帮助他们理解自己所处的环境。当数据是公开、免费、允许查看且可以对其自由辩论时，它是最美丽的。

这是一个关于奥克兰的Crimespotting项目(http：//oakland.crimespotting.org)的故事，它是位于旧金山市的Stamen Design设计公司(http：//stamen.com)的一个研究项目。Crimespotting项目（见图11-1）是为了响应当前奥克兰警察局的犯罪报告的一个应用——CrimeWatch(http：//gismaps.oaklandnet.com/crimewatch/）而开发的。正如很多其他项目那样，Crimespotting在最初构想时并没有具体的目标；它的产生是由挫折引起的，借助于基础、技术性的研究逐渐成熟起来，并最终因一起给人们带来创伤性的犯罪事件而变得众所周知，因为该犯罪事件引起了全国对该城市的广泛关注。Crimespotting项目的发展看起来经历了典型的项目发展曲线：经过有目标的酝酿，通常会发展演化成为成熟的信息化项目。该项目成为Stamen设计公司的顾问Ben Cerveny所谓的“信息化”的典型例子：世界上的数据正一点点地移动到互联网，并与某个开源工具链和方法论相互接触和碰撞。



图　11-1：Stamen Design设计公司的一个研究型项目——奥克兰市的Crimespotting项目标志

这个故事包含三个部分。首先，我们获得了奥克兰警察局的核心数据，将其抽取出来转化为更适合于“切片混合”操作的某种格式。其次，我们创建了一个动态网站，对数据进行公开，以便本地的居民可以找到并使用这些数据。最后，我们通过观察这个网站如何发挥作用，回顾最开始的设想，并答复公众的反馈意见。

背景

将现代的发布在线数据的方式应用于犯罪举报并不是什么新主意。当前对犯罪的关注可以追溯到记者/开发人员Adrian Holovaty在2005年研究的Chicago Crime项目(htp：//chicagocrime.org)。Chicago Crime项目是GMap混搭的一个早期例子，它利用来自其他站点的代码和数据打造了一个卓越的网站。在这个案例中，尚未文档化的GMap API(http：//code.google.com/apis/maps/）被看做发布芝加哥警察局的犯罪报告信息的基础组件。该警察局本身的网站是由文本驱动的，它刚好包括了每个报告的街道地址或十字路口。应用服务程序Holovaty在夜间收集报告数据，把它们发布到动态的、可缩放或“滑动”地图上。该服务作为早期G公司2005年在线映射的最佳实践，别出机杼，是个非常绝妙的方法技巧。GMap从原来的局限于只是画图和展示静态图片，转换成为可以展示不断滚动、可以无限回应的地理数据平台。

Chicago Crime项目不是唯一的。几乎是在同一期间，开发者Paul Rademacher开发了Housing Maps(http：//housingmaps.com)项目，它类似于Chicago Crime项目，是公寓出租数据和可视化浏览的组合体。在其之前一年，Michael Frumin和Jonah Perretti在New York艺术基金会Eyebeam创建了FundRace项目(htp：//fundrace.org)，实现了2004年总统竞选政治捐款数据的可视化。Rich Gibson、Schuyler Erle和Jo Walsh合著出版的《Mapping Hacks》（O'Reilly出版社）一书，是一本关于新的基于Web

制图的技术指南。针对这些事件，开展了很多在线活动，因为在地理信息上下文中，新的信息化数据集正在大刀阔斧地修改并重新发布。这一切所产生的更广泛的影响改变了人们对地图的期望：人们现在期望有新的、深层次的多规模交互方式；作家Steven Johnson称之为“长距离变焦”(Te Long Zoom)，一种包含游戏、电影和其他媒体的新视野。

GMap对于Web地图领域的影响是不可估量的，其中有两个原因。首先，GMap做出了采用了图片拼接的方法来发布地图的决策。这使得它可以不用重新渲染全部图片，减少了对服务器的压力，因而能够采用视觉上更复杂的制图技术。其他地图需要效果非常显著，才有资格与其竞争。其次，这个决策也意味着最后的可视化效果将变成由用户端的浏览器通过动态组合的方式完成。最近，Jesse James Garrett提出了一种关于浏览器的新兴技术——Ajax（异步JavaScript和XML)，这一术语后来变得尤为重要。Ajax技术说明了现代的浏览器已经有足够高的质量，且当它与标准足够兼容之后，人们开始重新开始对动态HTML和JavaScript感兴趣。采用这种方式后，原始数据的传输从制图展示中独立了出来，在一个单独的频道中完成。而在此之前，其采用的方式是在服务器端完成地图的生成、地址查找和确定方位等处理之后，将所有的数据打包成一张图片并将整张图片作为返回结果展示给Web访问者。

把地图分块、接口和数据组装放在客户端完成，这一点使得小开放团队创建非常复杂的基于地理的浏览应用变成可能，这也正是在我们和Web四年交互过程中所看到的。

所有这一切构成了从奥克兰现有的、获得商业授权的犯罪地图产品CrimeWatch中抽取可管理的、可重用的数据的努力的主要背景。它最初是由简单的技术好奇心所驱动，以及圣诞节的很多空闲时间无意中所带来的“背后一击”。

解决棘手问题

奥克兰的CrimeWatch是一个提供犯罪报告信息的应用，以图片的形式提供犯罪报告相关的原始制图和卡通式的图标。CrimeWatch为便于数据展示进行了专门优化，而且侧重于预测用户需求而不是提供原始材料。应用的用户体验是通过“向导小工具”(Wzard)来通知的，它是一个用户界面，提供给用户一系列的对话框，它们生成了一系列的步骤，通过特定的序列执行任务。CrimeWatch应用的必要步骤如下：

1.事件：选择事件的类型（一种或者多种）；

2.地点：在地址附近搜索，在可管理的范围内；或者是某个特征附近，如学校或者公园；

3.时间：在多长的时间范围内进行搜索。

CrimeWatch通过静态图片来响应，该图片以图表的形式显示各个报告。

点击这些图标可以获取更多的信息。第一次激发了对CrimeWatch的兴趣是在我开始思考某种方式用来颠倒服务端完成归并过程；从静态的图片开始，通过附加的地理信息来抽取犯罪报告信息：使用和那些其他地理软件系统兼容的经纬度值，通常被称为地理定位(golocation)。这种简单的识别问题很容易理解，而且在视觉特征抽取方面已经存在着很好的技术。

首先，我们需要获取一张图片进行处理。这过程实际上比看起来要

复杂得多，我们必须跳过一系列的“钩子”(hok)，“说服”服务器生成一张犯罪报告图片。CrimeWatch在服务器端保存了会话状态，因此有必要通过一个假的虚拟用户来模拟一套完整的向导互动模式：通过表单接受服务条款，然后通过交互向导继续完成各个步骤，在这些过程中保存HTTP Cookies和令牌(Tken)，然后对非标准的HTTP重定向进行正确地响应。重构这些步骤过程，从而生成有用的犯罪报告图像是该项目面临的第一个难题。幸运的是，客户端的HTTP代理软件Charles(http：//charlesproxy.com/）和Mozilla插件LiveHTTPHeaders(http：//livehttpheaders.mozdev.org/）简化了这种重构过程。解释中间HTML页面本身是通过使用网页刮屏(pge-scraping)库的方法，如Leonard Richardson的BeautifulSoup库(htp：//www.crummy.com/software/BeautifulSoup/）。BeautifulSoup库是为了使通常看到的HTML“标签”变得有意义，纠正了某些通用的问题如不正确的嵌入式标签或者部分标记，而且它允许我们读取HTML表格和JavaScript脚本命令，这些构成了一个完整的客户/服务器会话。

可以使用简单的Unix命令行工具如Shell脚本和网页访问工具cURL(http：//curl.netmirror.org/）来模拟第1版的刮屏过程。其关键是仔细检查浏览器和服务器之间的HTTP连接，寻找暴露出来的信息可以帮助重新构建交互。查看URL中包含的CGI变量和Post方式请求体是重构的第一步，它可以准确地表示接受了使用条件后的初始的会话连接。基于会话的应用如CrimeWatch大量地使用了Cookie来保存客户端的状态，

因此，利用HTTP库把Cookie打成Jar包是必需的。CrimeWatch还大量依赖于客户端的JavaScript框架，而不仅仅是简单的表单提交；此外，它还使用了额外的状态变量，这样中间响应页面必须通过HTML解析器和正则表达式才能够搜索隐藏在页面脚本中的详细信息。最后，由于很多这种老一代的Web应用是在跨浏览器的动态HTML被开发者普遍应用之前出现的，因此“欺骗”User-Agent头数据，假装是IE浏览器或者Mozilla火狐浏览器绕过User-Agent头数据通常是必要的；其他浏览器会显示不兼容报警信息，不返回任何数据。

在这个过程结束时，只剩下中等大小的图片位图，期望能够包含可识别的犯罪报告图标。通过第一轮抽取每个图标的像素位置很简单，但是过程很慢；对于图像中的每个可能的位置，把它的像素颜色和已知的图标做比较，一旦差异小于某个给定的阈值，就显示正的匹配度值。因为我们是在噪音相对较少的后台来处理预先指定的图标，这实际上是一种完全的“防弹”方法。CrimeWatch中使用的犯罪图标是独一无二的，而且即使部分被其他图标或者图片特征所屏蔽，但是还是可以很容易识别出来。我采用的检测这些图片的工具是NumPy(http：//numpy.scipy.org/），它是一个久经考验、功能强大的Python数组操作库。图11-2和图11-3显示了CrimeWatch的部分样本图片，突出显示了程序可识别的图标。

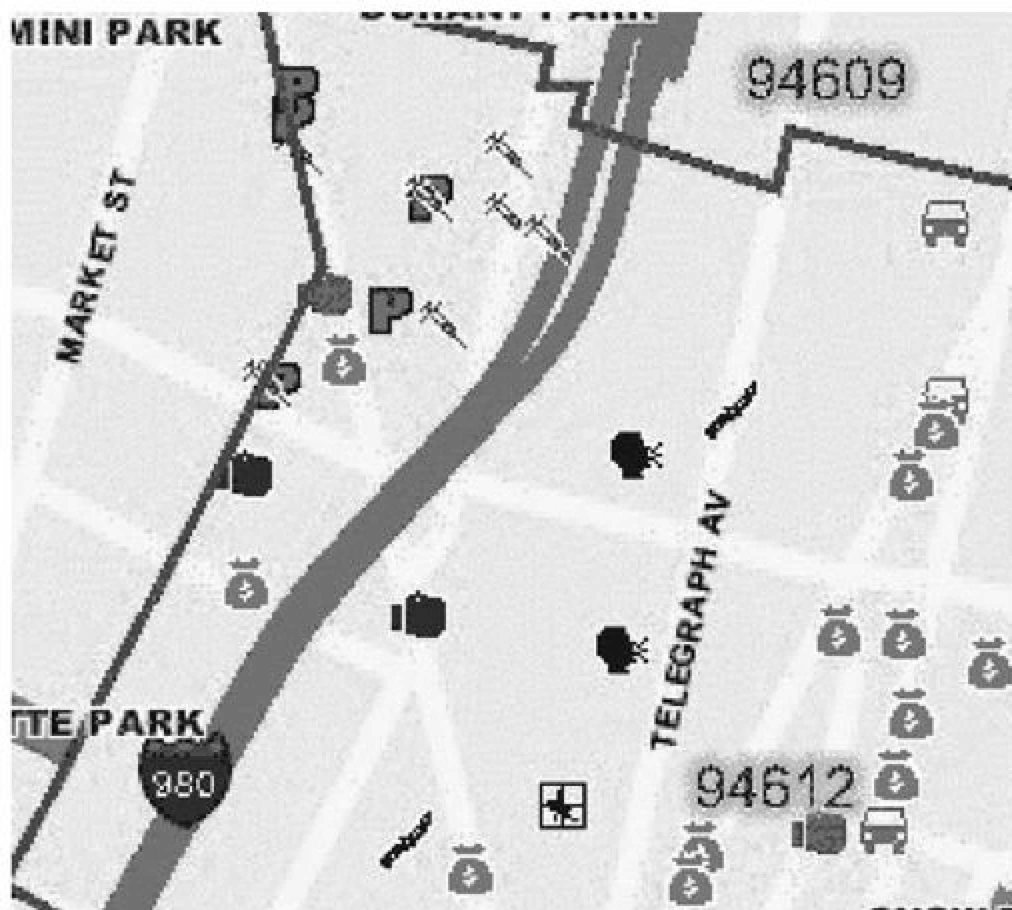


图 11-2：CrimeWatch的样本数据显示了盗窃、毒品、抢劫、盗窃车辆等犯罪活动（见彩图29）

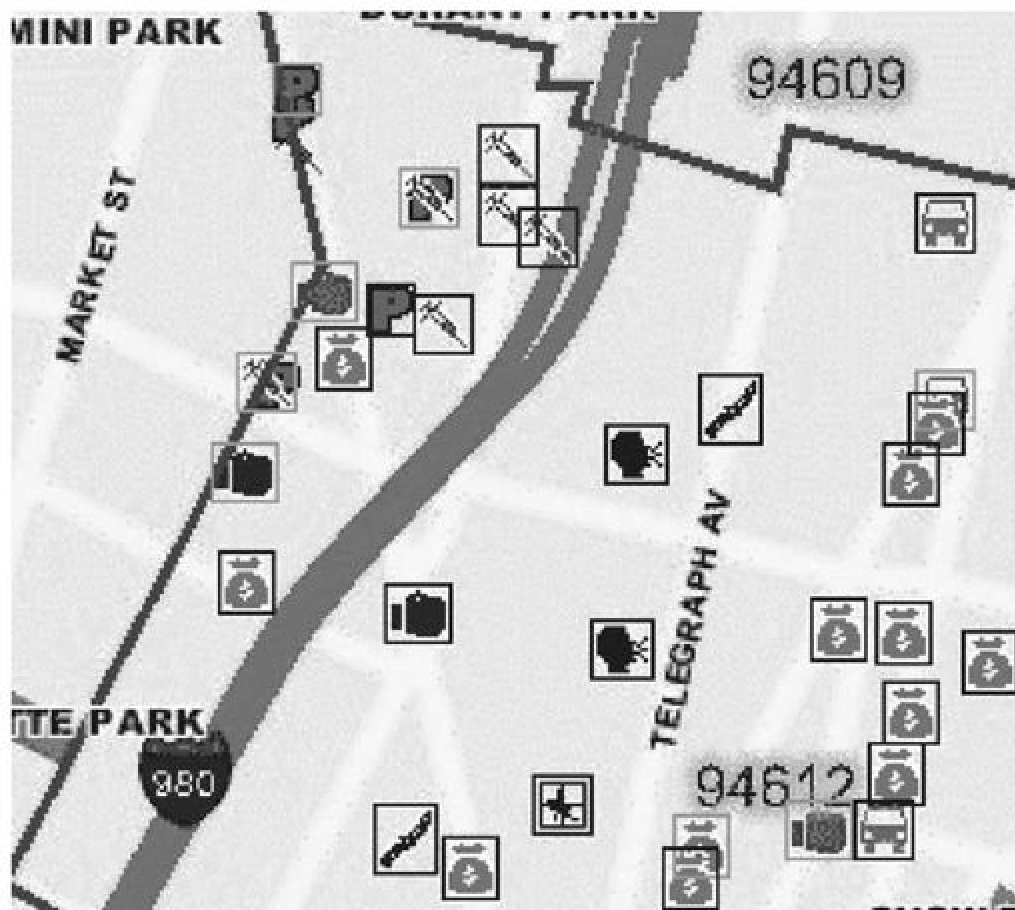


图　11-3：CrimeWatch相同的样本数据，突出显示了程序可识别的图标（见彩图30）

遗憾的是，蛮力方法在普通的CPU上很慢，但是通过对所需处理的各种地图的一些简单分析，也许可以对其速度进行一些优化。例如，很多犯罪报告图标有很显著的特征颜色：通常货币形状的绿色袋子表示盗窃，蓝色拳击手套表示普通伤害。一种简单的预处理步骤是通过代价较低的图片扫描来找到与其中一个期望的颜色接近的像素的位置，这种方式可以极大地降低代价高的全图标匹配的位置数目。图11-4用白色正好显示了图11-2的红色部分，它表示严重伤害可能发生的场所，可以看到有红色的拳击手套。



图　11-4：CrimeWatch的相同的样本图片，红色的部分用白色显示是为了更清晰地突出红色的拳击手套图标（表示严重伤害）

一个较为复杂的预处理步骤是通过一系列的扫描来搜索临近的几种特征图标颜色。例如，盗窃是通过黑白相间的破碎的玻璃小图标来显示的。一张经典的地图上包含很多的黑色和白色，但只有图标和文本会黑白相间。我们会找到所有和这两种颜色很相近的像素，并把高代价的搜索区域降低限制为只包含有限的候选像素。

每次犯罪报告的地理定位过程都需要根据在渲染地图中检测到的像素位置来确定位置。为了实现这一点，对于给定的一组输入，CrimeWatch总是返回可预测大小和位置的地图是有帮助的。例如，3号

警务服务区域(Plice Service Area)的地图需要总是覆盖相同的区域，而不用考虑当时实际的犯罪报告是集中于区域的某个角落还是扩散到所有地方。CrimeWatch提供的地图包含的底层地理特征如街道或海岸线，总是在相同的地方。对于每一种可能的地理布局方式，需要人工定位三个大间距的已知参照点。道路交叉口对于这点非常合适，因为在CrimeWatch上可以很容易地定位它们的（x,y）像素坐标，而且通过Simon Willison的GetLatLon应用(htp：//getlatlon.com)，也可以很容易地定位道路交叉口的地理位置（经纬度）。给奥克兰的6个警务服务区添加三个参照点相当于手工创建了18个已知的地理定位。该步骤只需要做一次：在每个服务区，所有能在地图上找到的图标，都可以利用简单的线性算法，通过和已知的参照点比较，计算出它们的地理位置。图11-5显示了商业区邮政编码的地图，它包含选定的三个地理参照点。定位了这三个点之后，就可以三角测量地图上的任何其他点的地理位置。



图　11-5：奥克兰商业区地图，为三角测量所显示的三个参照点（见彩图31）

唯一剩下的一件事就是模拟用户点击每个犯罪图标，收集关于犯罪报告的进一步详细信息，比如案例编号、每天的日期和时间以及简单的文本描述。其生成的最终产品是每天包含大约100个报告的数据库。该步骤面临的一个挑战是确定一个独特报告是由哪些构成的。我采用移动窗口来收集报告，这意味着每个报告都会被收集多次，而多个独立的报告可以通过警察局提供的唯一案例编号覆盖成一个。最后，我们使用了一组案例编号和文本描述，它们足够覆盖数据收集中绝大部分的不一致性。

实现上述功能的代码运行于事件驱动的网络引擎Twisted Python(http：//www.twistedmatrix.com/）上。采用这种引擎使得打开并保持一个长时间持续运行、使用CrimeWatch提供服务的模拟浏览器的会话变成可能。有了这些代码库，可以把脆弱的程序转变成每天晚上连续运行和收集数据的应用程序，而且结果数据也可以以表单方式公开展示。我们相信对于奥克兰市的居民而言，表单方式显示要比采用CrimeWatch显示显得更为友好。

晚上收集这些数据为最初8个月的收集和实验奠定了基础。每天晚上，我们会在13种类型的犯罪和6个警务服务区的全部组合上运行一个网页刮屏工具(Pge Scraper)。由于CrimeWatch的一次性设计，每个个人报告需要有它自己的请求/响应循环。我们还为每个步骤增加了很多延迟——在这个过程中，每两个步骤之间高达一分钟或者更多的延迟——因此不会由于过多的请求而使CrimeWatch服务器负载过高。在午夜之

后，将会启动单一进程运行，而且该程序通常可以持续6个小时或者更多。

一般来说，错误还是时常发生。CrimeWatch服务程序常常会变得非常“不理智”，勉强生成一个没有空间、时间或类型约束的地图，比如奥克兰过去三个月所有地区的所有犯罪。我们没有可靠的方式来检测这种情况，而且我们的数据库所报告的地理位置经常错乱。

在这个案例中，我们觉得偶尔的错误报告只是为了改进数据库浏览工具而付出的很小的代价。在2007年前半年，我们继续收集数据，通过可视化表示或者出版发表技术，周期性地发布一些小的实验。

公开数据

2007年8月2日，《Oakland Post》记者兼主编Chauncey Bailey在光天化日下被刺杀了，距离我所在的商业区的公寓仅隔几个区。虽然这个尚未公开的案例看起来是被当前Bailey正在调查的一群罪犯分子的政治谋杀的一个案例，但是它重新聚焦了全国对奥克兰暴力犯罪的关注。几乎同时，《Oakland Tribune》的Sean Connelly和Katy Newton发表了获奖作品《Not Just A Number》，它是一张奥克兰杀人犯的交互图(htp：//www.bayareanewsgroup.com/multimedia/iba/njn/）。Connelly和Newton对于该城市的谋杀统计幕后的故事尤其感兴趣。在《Not Just A Number》地图中，绝大部分的受害者先是通过面部照片来识别，它还努力联系受害者活着的家人、朋友和邻居在新闻报道中露面留名。我们对于提供一个服务来提供铁的事实和当前数据来补充这些故事非常感兴趣。

我们的预算包括三个人、两周的敏捷开发工作量：我们把收集的数据转化为可用于Web的服务，Stamen交互设计师Tom Carden使用Flash开发了可融入式的可视化接口，而富于创意的负责人Eric Rodenbeck预见了这种可视化方向及其伴随而至的语言所产生的效果。

发布信息优先级最高的是要显示所有方面。Crimespotting网站的主页是一张地图（见图11-6），而且该地图显示了从上周开始的所有犯罪报告。地图包含西奥克兰和商业区的绝大多数地方，包含图标化的Lake

Merritt，可用于视觉识别。熟悉的“滑动地图”(Sippy Map)的移动和缩放控制使得可以迅速地获取城镇的其他地方的信息，西北部是Berkeley/Emeryville，东北部是富饶的山脉和Piedmont，东南部是San Leandro及其以外地区。这种表示方式与奥克兰城市的官方网站City of Oakland发布的当前已有的通过向导的方式形成了鲜明的对比。现有的应用需要一些关于奥克兰的先验知识，而且它假定参观者寻找的是特定地方的犯罪信息，Crimespotting项目的“滑动地图”不需要先验知识或者特定的搜索议程表，相反地，它支持的是更有探索性的、更曲折的搜索行为。Peter Morville描述的是一种新兴的概念——“可寻性”(fndability)，它描述了在信息空间中进行定位的方法以及借助接口和描述使得数据实现自我描述的方法。动态的基于Web的地图在过去四年中经历了很多改变升级。在2005年，一家报纸网站采用GMap做试验，发现被测人员不知道他们可以拖动地图；现在像《New York Times》这样的组织会定期地把信息设计和表现方式在线公示。有了犯罪数据库，我们觉得通过创造数据优先的用户接口，使数据更具有“可寻性”是很重要的。数据首先意味着通过大范围的可视化概要表示是可能的，并且通过类型、时间或地理特征缩小了搜索结果范围。我们实现了“香型小工具”(Sented Widgets)，由加州大学伯克利分校的研究人员Wesley Willett、Jeffrey Heer和Maneesh Agrawala在2007年嵌入式可视化方面的论文(htp：//vis.berkeley.edu/papers/scented_widgets/2007-ScentedWidgets-InfoVis.pdf)中提出：



图　11-6：奥克兰Crimespotting项目的主页显示了从上周开始的犯罪报告地图（见彩图32）

虽然有效的信息“气息”线索可能是基于底层的信息内容（如在Web超链接的文本描述了链接文档的内容，这就是“气息”），其他线索可能涉及各种形式的元数据，包括使用模式。在物理世界，我们通常以浏览的方式对待其他人的活动。当发现一群人聚集在一起的时候，我们可能走过去以便观察确定引起大家兴趣的根源到底是什么。或者，我们可能故意避开人群或者陈腐的途径，选择“人迹罕至”的方式来探索鲜为人知的兴趣点。在信息空间环境中，这种社会性的浏览会把我们的注意力吸引到兴趣的热点或者尚未探索的区域。

Crimespotting地图左下角的主接口的时间选择器界面显示了报告的犯罪随着时间变化的条形图（见图11-7），虽然在右下层类型选择器包含了谨慎的工具提示信息，显示当前选择时间范围的每种报告类型的总数（见图11-8）。这两种选择器都包含双重功能：过滤和反馈。特别是，时间选择器是受到博客统计包Measure Map(http：//measuremap.com/）的一个类似功能启发，Measure Map是Adaptive Path公司的Jeffrey Veen设计的，后来变成G公司的分析产品。而反过来，Measure Map的时间滑动轴是受Flickr的界面特征所启发的，因此，坦白说，这实际上是模仿。我们自己加强的是不同条形栏的颜色区分，并显示已经包含的数据的日期（深色）和不包含的数据的日期（浅色）。

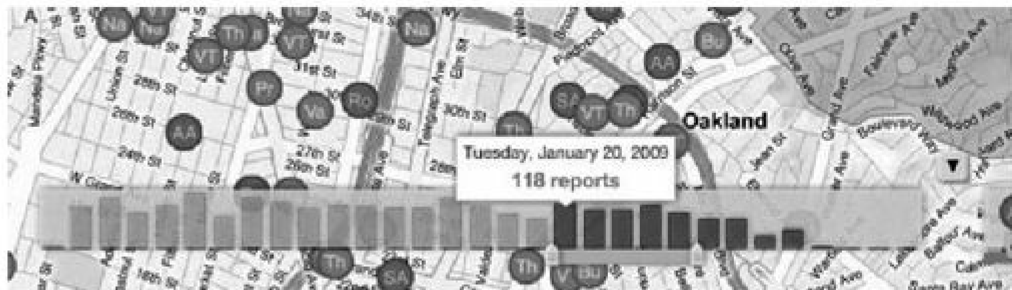


图　11-7：Crimespotting的主地图上的时间选择器界面（见彩图33）

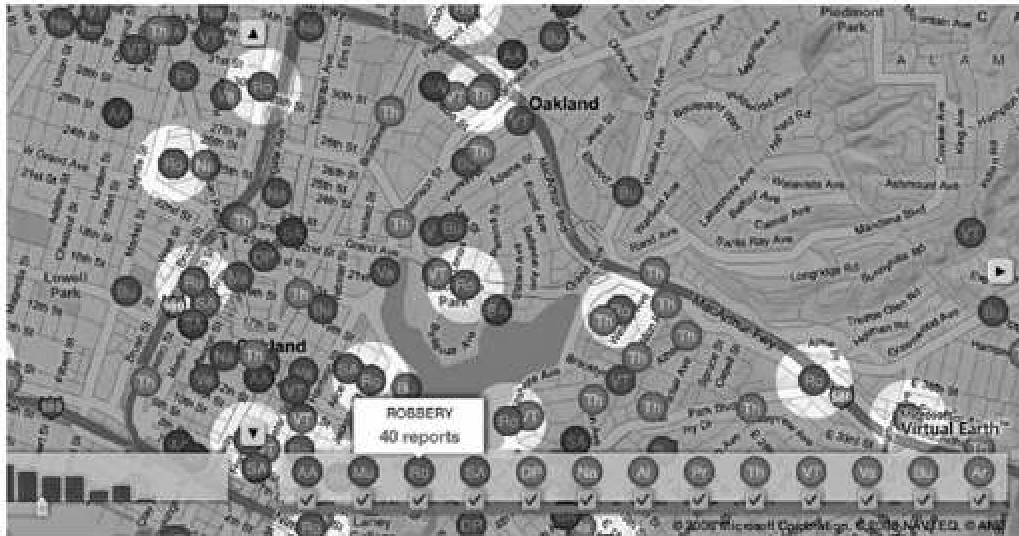


图 11-8：类型选择器显示了在选定时间范围内的每种报告类型的总数（见彩图34）

在图11-8中，有一个水平滑动轴显示所有东西，而它就是信息过载。我们已经介绍了一种可视化报告类型过滤的方式，它是受到苹果公司的Max OS X系统偏好的聚光灯特征所启发：当在地图中存在特定的犯罪报告，或者通过鼠标在类型选择器上多滞留几秒钟，界面颜色就变深，使得地图中高亮显示的区域围绕匹配该类型的报告。抢劫可能会由于不同的犯罪类型被覆盖，导致无法在地图上显示，但是可以通过聚焦显示方式展现和访问它。

设计过程中的意外收获使我们更清楚地理解了数据专业化的哪些方面可以成为应用界面产品。考虑到Flash在视觉特技方面相当成熟，我们选择它作为实现环境，而且及时实现了我们自己必须实现的滑动地图交互代码，而不是依赖于很多已有可选的JavaScript库，如OpenLayers(http：//www.openlayers.org)。

平移、缩放地图交互看起来是可以应用于其他项目的一个有用的功能点，因此在犯罪数据显示的早期工作成果是生成了独立的通过BSD授权的软件库Modest Maps(http：//www.modestmaps.com/）。Modest Maps在数据显示和交互隐喻之间的功能有一个清晰的突破，而且它把地图相关的代码库独立出来，这有助于很多不相关的项目的敏捷开发，这些项目有一些源于Stamen，但是很多是来自于外界的设计和开发。

第二优先级是为我们收集的数据生成一个公开、可共享的地址空间。通常来说，在Crimespotting项目中只有一些URL“气息”：

·地图视图http：//oakland.crimespotting.org，更大的视图在http：//oakland.crimespotting.org/map。

·报告列表视图，如http：//oakland.crimespotting.org/crimes,http：//oakland.crimespotting.org/cr
01-09和http：//oakland.crimespotting.org/crimes/2009-01-09/Robbery。

·个人报告视图，如http：//oakland.crimespotting.org/crime/2009-01-09/Robbery/113569。

·巡警视图，如http：//oakland.crimespotting.org/beat/04X。

绝大多数URL是在把它和内容关联起来之前设计的。特别地，这些URL需要和设计师Matt Biddulph在2005年发表的演讲“Designing Data For Reuse”(http：//www.hackdiary.com/slides/xtech2005/）的思想一致：可读性、建设性、易于破解、不透明、永恒和经典。我们包含地址的一个层次结构，它在大声朗读时富有韵律：“1月9日抢劫”，“04X巡警”等。其

中存在潜在的二义性——例如时间在前“/犯罪/2009-01-09/抢劫”和类型在前“/犯罪/抢劫/2009-01-09”或者单数形式的“/犯罪(cime)/抢劫”和复数形式的“/多重犯罪(cimes)/抢劫”——我们把这些HTTP请求重定向到它们自己相应的表单。该重定向通过确保一个人给出的某天的盗窃清单和另一个人给出的能够完全匹配，这样URL就变得更具有共享性。通过个人给出的URL，其不足之处在于它的地址结尾存在一个数值关键字。PostgreSQL开发人员Josh Berkus非常憎恶这种关键字，在他的“Primary Keyvil”的系列文章中对此做了详细的描述(htp：//it.toolbox.com/blogs/database-soup/primary-keyvil-part-i-7327）：

没有多久（大约两个月），我们就发现使用“id”作为唯一的不重复字段存在严重问题。我们在日历安排上有很多的听证会，它们具有相同的摘要、在同一天或者在同一个地方发生。这两个听证会是相同的还是不同的？我们无法确定。问题的本质是自动id列不包含其关联记录的信息，而且不包含该记录的任何信息。记录本身可能是重复的，也可能是唯一的，而如果有人无知删除了外键约束，重复的记录就丢失了。

我们包含这些关键字的理由是奥克兰警察局保存记录的方式相当松散。虽然每个报告都有一个案例编号，但这些编号通常是在不同报告之间共享，而且看起来把一组个人的指控连接为单个更广泛的事故。一个极端的例子是编号为08-056061的案例(htp：//oakland.crimespotting.org/crime/2008-08-01/murder/93014），它关联了9个在2008年8月某个晚上发生的谋杀、盗窃和严重伤害报告。我们

已经使用案例编号和文本描述（如“ASSAULT W/SEMIAUTOMATIC FIREARM ON PEACE OFFICER/FIREFIGHTER”）作为唯一标识符来解决这个问题，但这种方式导致URL太长，可读性很低，因此使用数值ID来代替。

对URL的这种关注的结果是把在线的犯罪信息转化为社会对象。在CrimeWatch上，引用报告需要对要采取的动作使用过程化的描述：采用向导小工具(Wzard)，可以选择这个，按住那个，点击这里等；找到关于某个特定报告的特定信息需要大约10多次的独立点击。URL公开之后，其本身也是犯罪信息的完整描述。Leonard Richardson认为地址或者统一资源标识符URI是导致WWW在20世纪取代了其他流行的网络协议的主流技术。在Richardson 2008年的精彩演讲“Justice Will Take Us Millions Of Intricate Moves”(htp：//www.crummy.com/writing/speaking/2008-QCon/）中，他认为Web主要有三种技术构成：定位资源的URI协议、传输资源的HTTP协议和使客户端软件能够理解并知道如何处理资源的HTML。这三者都是至关重要的组成部分。特别地，URI设计受到人们的广泛关注，但实际上是包含链接和表单的又老又慢的HTML使得相互连接的Web真正变成可能。HTML是对Roy T.Fielding在2000年博士论文中提出的“表示状态传输(Rpresentational State Transfer,REST)作为架构方式”(htp：//www.ics.uci.edu/～fielding/pubs/dissertation/top.htm)想法的至关重要的解释。只要可能，我们就遵循这些想法，保持Crimespotting的

交互的Falsh展示部分严格基于基础的、1993年的Web页面的格式。我们的API为Flash、RSS和Atom的Feed读者提供XML格式的数据，为电子表格提供CSV格式的数据；所有核心的使用信息方式组成了一套完整的API。

2007年8月发布的产品中囊括了这里描述的所有概念，同时依赖于CrimeWatch在夜间执行的代价很高的刮页操作。我们确定市政府肯定有人最终会注意到这一点并对它表示不满，但是八个月来系统一直运行很顺利，这让我们产生了一种安全的错觉。

重新回顾

发布后不久，我们的网页刮屏程序的运行负荷开始超出能够容忍的底线。即使使用通用的浏览器，似乎不论等待多长时间，都无法访问CrimeWatch网站。城市信息技术部相关人员的谈话，暗示我们一旦公众注意到这些访问问题，该网站就会不受欢迎。他们提供了通过官方访问这些数据的一些线索，但是他们那种作风的方式处理很慢，没有任何即时收效。我们很遗憾地关闭了该网站，花了几个月考虑优化程序以及改进策略，使它能够重新运转起来。在那时，我们尝试了两种思路来优化该系统，同时还开发实现了一个新功能。对系统的两种优化最终都未成功，而新开发的功能则对外发布了。这次修改过程的一个产出是生成更集中、实用的最终数据展现方式。

在思考如何最佳表示本地犯罪的影响时，我和Adam Greenfield的谈话激发了这种想法："暴力是发生在某个地方的一种力。"在周边社区的谋杀或者抢劫所带来的长期影响可以被看做一个环（见图11-9）。这种想法的在我的大脑中产生的初始模型是一个空间-时间半球，球的空间半径约0.25里，时间半径是一周。这种可视化显示方式将是：从一个点开始，随着时间轴逐渐靠近精确的事件发生时间而变大为一个有色圆。Greenfild建议将半球转换为一对圆锥：犯罪的时间点是两个圆锥的共同的顶点，在时间轴上以这个点为起点向前和向后各有一个"光锥"(lght cone)。其视觉显示效果将是一个大的散开的圆圈随着逐渐接近事件发

生的精确时间而逐渐清晰化，并最终汇聚为一个微小的点。这种展现方式的理念可能更适合于展示对象的随机属性。犯罪的潜在可能性可能很广，甚至可能涵盖整个社区。随着事件的曝光，这种潜在的危害逐渐聚焦为一个点-即某个邻居成为受害者，然后随着消息传播开来，个人安全感也消失了。

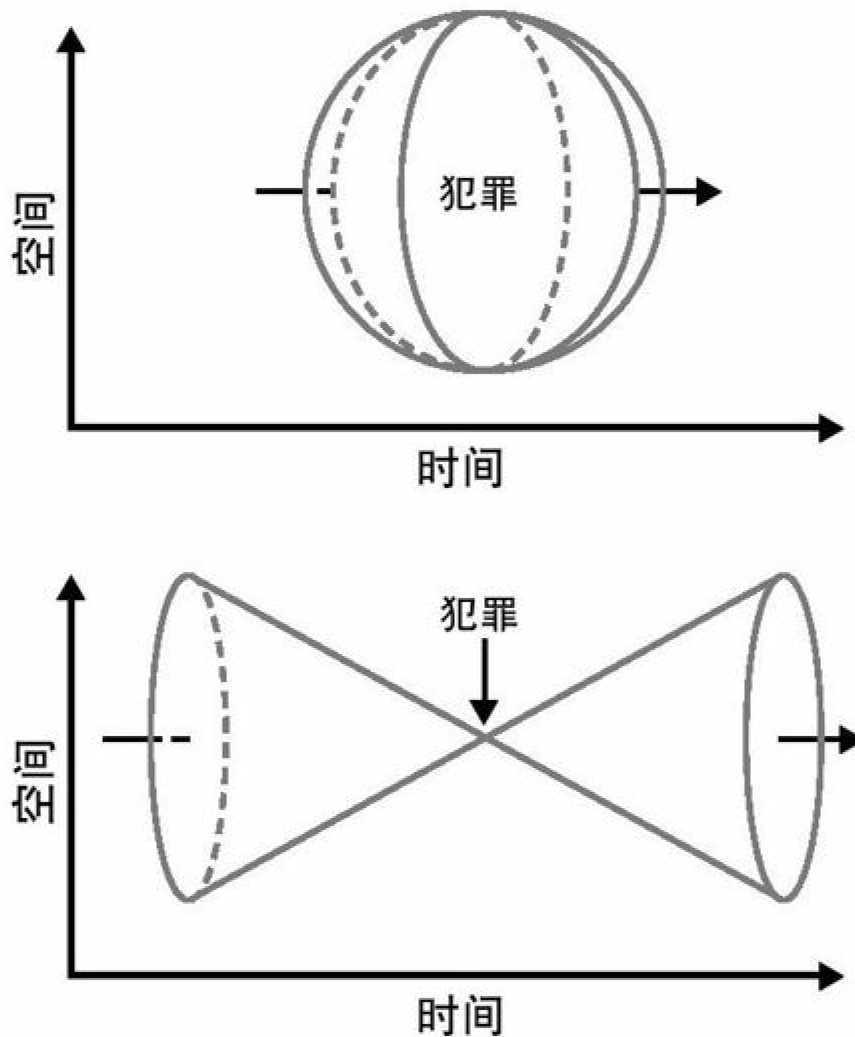


图 11-9：设想的一起谋杀或抢劫对周边社区的长期影响的两种方式

我们生成了很多交互式地图来探索这种以圆锥作为隐喻的方式，并

发现了一些有趣的现象。其中一个现象是某些类型的报告存在依赖于其实施方式的唯一视觉签名。卖淫就是其中的一个特例。在之前的案例中，我们展现的绝大多数报告都是事件驱动的-受害者给我们打电话，而卖淫则是由警察局的决策和安排的严打所驱动的。通常，在前端展示上，关于卖淫方面，好几周都是风平浪静，而后会突发产生关于San Pablo Avenue或International Boulevard的迅猛的、翻天覆地的信息。遗憾的是，这种圆锥式的展示方式太怪异了，以致在主要的网站上无法使用。而在地图上，通过时间遍历的方法非常新颖，而且对我们来说，使报告展示和时间的关系控制尽量明朗是很重要的。圆锥式展示方式将还需要进一步试验。

在我们关闭网站时期，我们考虑到另一种可能的改进是采用分布式的网页刮屏方式(pge scraping)。正常的收集过程容易受到中断的原因是所有请求都来源于相同的网络地址，这使得这些请求在需要时很容易被阻塞。我们试验了一种分布式的模型，它是火狐浏览器的一个现成的插件，主要通过JavaScript控制。我们希望有足够多理解技术的访问者会愿意下载浏览器搜索框，帮助收集数据。这种方式，和通常的网站使用方式截然不同，发到CrimeWatch服务器的请求将会在每天不定时地被分散到很多访问IP地址中。这种处理过程的优点是在最后增加了人工错误纠正。中间件刮页过程的最后页面中包含了用户帮助收集的所有报告的总体概述，还可能标注某些匹配为不正确的。

在这期间开发的一个新功能是巡警区专用(bat-specific)页面，比如在

商业地区和Lake Merritt地区之间的商业和居住区域：

http：//oakland.crimespotting.org/beat/04X（见图11-10）。当我们最初开发该服务时，我们有意识地决定忽略在CrimeWatch中出现的管理部门。警务服务区域、城市委员会区、邮递区、巡警区对我们而言都只是对定位的一个干扰。在发布后，我们很快明白了我们对巡警区的理解是错误的。用户告诉我们公民和警察局间的沟通主要的渠道是巡警区的警官，这些警官各自有特定的巡查区域且会定期与当地社区的居民开会沟通。报告按照巡警区进行划分是非常重要的，因为它匹配了任何给定的警官关注的区域和责任。另外，巡警区的边界通常是该城市的明显的物理特征，主要街道、小溪、高速公路和铁路都是作为社区的自我标识的关键特征。巡警区页面显示的是该区域的静态全局概要地图，以及对于习惯于电子表格软件的非技术人员来说，它是比较有用的一部分API。在这个功能中，我们收到的最终的反馈是非常宝贵的。居民说："我们下周有Beat1X NCPC（社区犯罪预防委员会）会议……我可以比那些老警察(OD)准备得更充分……到目前为止，他们几乎没有将统计数据分享给我们。"

火狐浏览器插件及其关联的Web服务控制器完成后，正在计划做一些有限的、实验性的首次展示的时候，奥克兰城市官方报纸Ci ty of Oakland告诉我们，他们将每天晚上给我们提供一个包含合适的街道地址和岔路口的完整的全城范围的犯罪报告信息列表。从2008年1月开始，一直持续到今天，由于市政创建者以及与我们一起工作的数据管理

员帮助，我们的数据收集过程已经从冗余、易错的事件驱动方式转化为敏捷的处理方式。最终，在奥克兰CTO Bob Glaze、项目经理Ahsan Baig和City的Julian Ware和Andrew Wang的帮助下，我们获得了在夜间生成的官方犯罪报告数据的Microsoft Excel电子表格。改进后的系统和之前的犹如天壤之别：原来需要几个小时的数据收集和处理，现在只需几分钟。定位信息也变得更加真实可靠，地图信息详细到了街区级别，突出街道名称和叉路口特征，取代了原有的彩色图标。

结束语

我们是否成功地维护了和最初提出的“美的理念”(te ideals of beauty)一致的数据服务？Crimespotting项目的犯罪报告数据生动有趣，而且定期激励关注的居民发邮件，并支持邮件告警订阅服务的人数达到几百人。犯罪对任何城市居民，都是一项严重的问题，而它对于像奥克兰这样因犯罪事件而昭著的城市来说，犯罪对于人们的生活影响尤其重要。我们发布的数据有用吗？



图　11-10：巡警区专用页面允许居民为在当地巡逻的警官提供反馈（见彩图35）

我们定期收到来自使用新闻订阅的居民的来信，并且向那些居住在事件发生地的居民发送邮件告警或者为其推荐新的居住社区。我们是否足够自由或者公开？

所有的网站信息都是可以通过适合于很多种大范围的技术专业性的形式获得，从简单的每日邮件订阅或者电子表单到更复杂的新闻Feed订阅方式或者基于XML的API。该项目是有收获的、成功的，它创造了我们认为对本地居民最有利的数据服务。

通过创建并继续维持奥克兰的Crimespotting项目，我们在社会和政治方面学到了重要的一课。为了满足居民相互关联、相互通信的期望，城市和政府信息正逐渐迁移到网络上。随之而来的是关于“数据是公共产品”的理念，正如FortiusOne公司总裁Sean Go rman 2009年早期在一个重要的博客上发布的帖子所解释的(htp：//blog.fortiusone.com/2009/01/28/data-is-the-public-good-data-is-the-infrastructure-data-is-the-stimulus/）。为“专业消费者”(posumer)提供的数据可视化和分析工具的增长使得数据变得更加重要，因而可预测的、值得信任的网上原始数据，则更倾向于过于受束缚的基于Web的用户界面。同时，人们也逐渐精心设计关于发布原始数据的新兴的实践和约定，而且在很多情况下，为了把注意力转移到问题的形式或者可达性上，那些感兴趣的、有能力的非项目成员的介入是必需的。事实胜于雄

辩，而主动为公众的利益而公开和改进数据所做出的改变是最有效的。

第12章　Sense.us的设计

Jeffrey Heer

必须承认的是，我不信任美丽的数据，至少在没有上下文的情况下，我是不信任它的。第二次世界大战之前，荷兰政府收集了详细的民事记录，登记备案了荷兰居民的人口统计信息。这种做法的出发点很好，收集人们的注册信息来为政府服务管理提供信息。然而，德国入侵后，这些数据被用于有效地锁定少数族裔(Coes 2006）。1940年前荷兰还有大约14万的犹太人，而其中只有大约3.5万人幸存了下来。

虽然这种例子也许有点极端，但对我来说，这种发人深省的故事说明了一个基本的道理：数据之“美”取决于这些数据如何为人所用。数据蕴涵着帮助人们改进对事物的理解，并做出更优决策的潜能，因此数据是“美丽”的。要让数据实现其价值，需要做到收集并保护正确的数据，让合适的用户访问并理解使用它们。AOL公布的匿名性不足的查询搜索数据所造成的骚动是近期关于数据保护失败的一个实例。

幸运的是，绝大多数例子并不像这两个故事这么悲惨。实际上，更普遍的现象是数据都被浪费了：把数据收集并存储在数据仓库中——有时付出的基础设施代价很高——但还是尚未被充分利用。对于公司和政府，不断衰竭的数据意味着机会的错失以及很低的投资回报率。数据的价值和人们从中抽取出有价值信息并用它来指导实际行动的能力成正比。

然而有些自相矛盾的是，一些数据集拥有的“美丽”（或者潜在美）要高于其他数据。显然，选择收集什么数据以及如何设计数据存储的基础设施、模式、访问机制决定了在避免数据给人们带来伤害的同时，激发数据给人们带来信息和启发的潜能。但是，攀越美丽巅峰的“最后一里”(lst mile)是人类-信息相互交互的问题，即为了支持对数据的分析和通信，人们应该如何展示和探索数据。

本章将介绍一个交互式可视化应用的案例研究，以帮助培养人们实际应用美丽的数据：sense.us的设计，它是一个对美国的150年人口普查数据进行协同式探索和理解(snse-making)的Web应用。我将谈到建立庞大的、政府部门收集的数据集的步骤——美国人口普查——而且通过一组交互可视化方式使所有用户都可以访问这些数据集。我还将描述我们设计的分享和讨论机制，该机制的设计目的是吸引数据的浏览者们在对社会的理解和思考方面组成一个群体。我们的目标是通过培养集体数据分析来实现数据的潜在之美。

可视化和社会数据分析

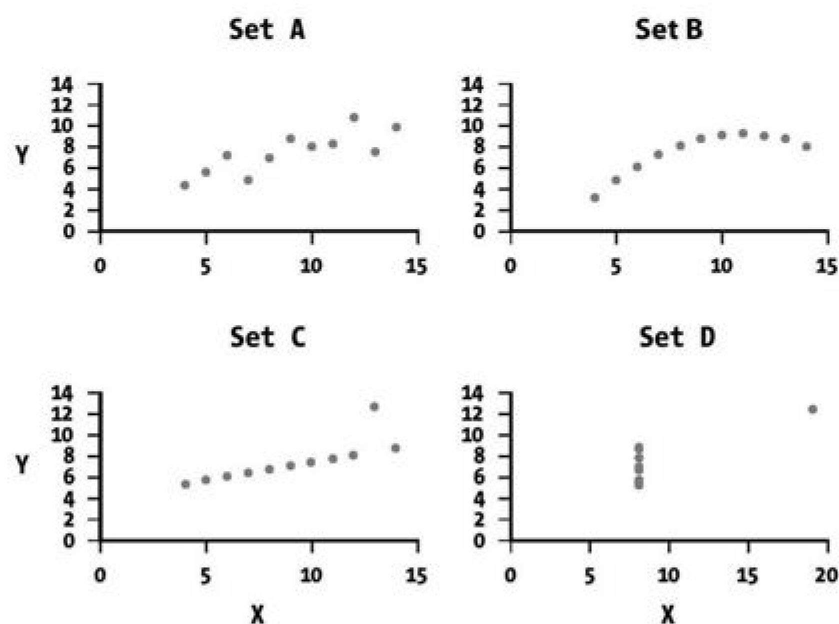
可视化通常用于构建数据的涵义来帮助理解这些数据，探索这些数据并交流结果发现。为了处理可视化信息，人类神经系统的很大一部分已经得到进化，在人类大脑中，超过70%的神经末梢和40%的皮质在视觉处理中相互关联(Wre 2004）。可视化设计充分利用了这种视觉处理系统的能力，使人们可以更容易理解并发现蕴藏于数据中的趋势、模式和异常信息。

需要注意的是，这不是手工制作一个“花哨的图形”的问题。通常，一个简单的表格或条形图(sns 3D虚饰frills和镜面高光）能提供很好的演示效果。这其中的特技在于为现有的数据和任务选择合适的视觉展现形式。

一个富有启发性的例子是Anscombe的“四重检测”(Qartet)，它采用由统计学家Francis Anscombe收集的四类数据集来说明数据可视化的重要性(Ascombe 1973）。使用通用的统计学描述方式，这四类数据集看起来似乎完全一样（见图12-1）。但是，对这些数据的绘图展示就可以马上揭示出这些数据集之间的差异。

还有其他作家详细地描述了有效的视觉设计是如何辅助人们理解、沟通和制定决策。Tufte（1997）提出了著名的论断，如果当时工程师创造了更好的对火箭缺陷数据的视觉描述，挑战者号航天飞机的灾难可能可以避免（虽然该论断也饱受争议，见Robison等2002）。

可视化研究人员列出了“视觉变量”(vsual variables)的空间（如位置、长度、面积、形状和色彩）这些变量可以用于在可视化显示中对数据编码(Brtin 1967，Cart等1999）。他们还研究了当这些视觉变量应用于不同类型的数据，如范围的（名字）、顺序的（基于排序有序的）和定量的（数值的）数据时，人们对其解码可以达到的精确程度。举个例子，如在条形栏或散点图中所使用的空间位置，有助于对这些数据的每一种类型进行解码；当采用色彩对不同的标签进行分类时，它的区分度很高，而用于描述数值时，它的区分度则很低。



Set A		Set B		Set C		Set D	
X	Y	X	Y	X	Y	X	Y
10	8.04	10	9.14	10	7.46	8	6.58
8	6.95	8	8.14	8	6.77	8	5.76
13	7.58	13	8.74	13	12.74	8	7.71
9	8.81	9	8.77	9	7.11	8	8.84
11	8.33	11	9.26	11	7.81	8	8.47
14	9.96	14	8.1	14	8.84	8	7.04
6	7.24	6	6.13	6	6.08	8	5.25
4	4.26	4	3.1	4	5.39	19	12.5
12	10.84	12	9.11	12	8.15	8	5.56
7	4.82	7	7.26	7	6.42	8	7.91
5	5.68	5	4.74	5	5.73	8	6.89

Summary statistics

$\mu_X = 9.0$ $\sigma_X = 3.317$

$\mu_Y = 7.5$ $\sigma_Y = 2.03$

Linear regression

$Y = 3 + 0.5X$

$R^2 = 0.67$

图　12-1：Anscombe的“四重检测”(Qartet)，包含统计学上相似的数据集来说明如何利用可视化帮助人们理解

本着这种思想，绝大多数可视化集中于研究可视化应用的感性和认知方面，通常是在单用户交互系统的上下文中。然而，在实践中，可视化分析通常是一个社会过程。人们可能对于如何理解数据有不同的见

解，可能会提供上下文知识来帮助加深理解。由于参与者达成共识，他们可以从同龄中互相学习。此外，有些数据集很大，由一个人完全深入探索是不可能的。这意味着为了全面支持有意义的决策(snse-making)，可视化应该也支持社会交互。

基于这些观点，我和同事Martin Watternberg以及IBM研究院的Fernanda Viégas开始调查用于可视化数据的用户界面如何可以更好地促进“可视化的社交生活”。我们开始启动研究项目，设计和实现网站sense.us，致力于形成人口统计数据的团体探索。该网站提供了一套美国过去150年的人口统计数据的可视化的解决方案，以及促进面向团体的数据分析的协作机制。

在本章的剩余部分，我将分享sense.us网站的设计过程，包括：我们如何选择和处理数据，开发一套可视化方案，并设计协作特征来促进社交数据分析。最后，观察人们如何通过该系统协作构建数据。

数据

Martin、Fernanda和我以研究人员的心态开始做这个项目：我们希望能够理解在可视化分析过程中，应该如何最好地支持社会交互。我们对数据集的选择不是预先确定的。但是，很显然，好的数据领域的数据集将会满足某些特定的属性：我们想要大规模的、真实世界的数据集，它们和普通的大众相关，而且数据丰富，可以保证很多不同的分析。根据这些标准，人口普查统计的数据看起来很理想。我还一直对促使人口统计数据更公开、可访问感兴趣：我相信这是重要的一个方面，通过它我们可以更好地理解自己和历史。

我通过美国人口普查局的网站(http：//www.census.gov)开始搜索数据。事实证明，这种方式收效甚微。人口普查局在各种聚集层次提供了很多不同维度的数据集（如通过邮政编码、都市区域），但是只在最近几年人口普查才有丰富的数据。我还意识到自己在该网站搜索数据时不够集中深入。人口普查数据在过去几十年如何收集和建模的来龙去脉，还亟待进一步了解。例如，人口普查局在收集数据过程中使用的问题以及涉及的范围在过去几十年在不断变化，意味着即使该网站包含某个人每年的数据，它也并不能保证这些数据能够很容易具有比较性。

一般情况下，人们在对某个数据领域有了最基本的熟练程度之前，不应该一下子扎进可视化设计。因此，我的下一个步骤是访问各个领域的专家们：加州大学伯克利分校的社会学和人口学部门的同事，那时我

是该部门的研究生。通过和他们的讨论，我对人口普查是如何工作以及人口学家使用哪些数据源研究人口有了更深的理解。在这个过程中，有人为我介绍了一个宝贵的资源：由明尼苏达大学人口中心(htp：//www.ipums.org)维护的综合公用微观数据序列(Itegrated Public Use Microdata Series,IPUMS)数据库。IPUMS-USA数据库包含了1850～2000年的美国人口普查数据。其普查周期是10年，只有1890年例外，没有普查数据；在1921年这些数据记录在商业大厦(Cmmerce Building)被大火摧毁。IPUMS数据包含了每10年的代表性样本数据，占这10年的人口普查记录的总样本数据的1%或5%。每条记录代表人口样本的某个特征人物。在某些情况下，有某些特征的人物和家庭被多次代表，因而记录需要和不同的权值关联。

IPUMS-USA数据库特别吸引人之处在于IPUMS项目为所有的人口统计学变量开发了统一的代码和文档，这有助于随着时间分析变化。这种“和谐化”是一种宝贵的服务，使得能够进行比较性分析，而且通过扩展，可以获取一些新的洞察。然而，把分离的数据应用于共享模式的过程不可避免地引入了人工干预，该问题在后面还会出现。

诚如之前所述，IPUMS-USA数据库包含413个人口统计学变量，其涉及的范围从普通的领域如性别、年龄、种族、婚姻状态和职业，直到多少家庭包含洗衣机、烘干机、抽水马桶和电视（见图12-2）。在很多情况下，一些变量只在某几年有数据记录，而在其他年份，这些变量完全没有被测量。

Demographic Variables (Person) top

Detailed Version	General Version	Variable	Label	Case Selection	Attach Variables	1850	1860	1870	1880	1900	1910	1920	1930	1940	1950	1960	1970	1980	1990	2000
☐	☐	RELATE	Relationship to household head	☐	☐	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
☐		IMPREL	Imputed relationship to household head	☐	☐	x	x	x	x	x	x	x	x	.	.	.	.	.	.	.
☐		AGE	Age	☐	☐	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
☐		SEX	Sex	☐	☐	x	x	x	x	x	x	x	x	x	x	x	x	x	x	x
☐		MARST	Marital status	☐	☐	.	.	.	x	x	x	x	x	x	x	x	x	x	x	x
☐		AGEMONTH	Age in months	☐	☐	x	x	x	x	x	x	x	x	x	.	.	.	.	.	.
☐		BIRTHMO	Month of birth		☐	.	.	x	x	x	x	.	.	.	x	.	.	.	.	.
☐		BIRTHQTR	Quarter of birth	☐	☐	.	.	x	x	.	.	.	.	.	x	x	x	x	.	.
☐		BIRTHYR	Year of birth		☐	.	.	.	.	x	x	.	.	.	.	.	.	.	.	.
☐		AGEMARR	Age at first marriage	☐	☐	.	.	.	.	.	.	.	x	SL	.	x	M	x	.	.
☐		DURMARR	Duration of current marital status	☐	☐	.	.	.	.	x	x	.	.	.	SL	.	.	.	.	.
☐		MARRNO	Times married		☐	.	.	.	.	.	x	.	.	SL	SL	x	M	x	.	.
☐		MARRINYR	Married within the past year	☐	☐	x	x	x	x	.	.	.	.	.	.	.	.	.	.	.
☐		MARRMONTH	Month married	☐	☐	.	.	x	.	.	.	.	.	.	.	.	.	.	.	.
☐		MARRQTR	Quarter of first marriage	☐	☐	.	.	.	.	.	.	.	.	.	.	x	M	x	.	.
☐		WIDOW	Marriage ended by death	☐	☐	.	.	.	.	.	.	.	.	.	.	.	M	x	.	.
☐		CHBORN	Children ever born	☐	☐	.	.	.	.	x	x	.	.	SL	SL	x	x	x	x	.
☐		CHSURV	Children surviving	☐	☐	.	.	.	.	x	x	.	.	.	.	.	.	.	.	.
☐	All Demographic Variables																			

图　12-2：在IPUMS-USA数据库中可访问的人口统计学变量的摘要

IPUMS项目的宗旨是“只用它从善，不作恶”(ue it for good,never for evil)。幸运的是，在实施该箴言的过程中，超出了人们在下载数据摘要时必须点击的复选框。为了保护个人隐私，有些数据的可访问性受到了限制。举个例子，不包含宗教从属关系，而且详细的人口统计数据的可访问性也受到很大限制，尤其是对于人口密度低的地区。

我们决定采用IPUMS-USA作为sense.us应用的主要数据来源。使用IPUMS Web界面，我们首先选择了1850～2000年的样本数据（不包括1890年），然后选择了需要抽取的一组变量。绝大多数变量只是所有人口统计年份的一个子集。为了对长期的变化进行可视化探索，我们选择了一组变量，它们至少在一个世纪内都是可访问的。这组变量由22个变量组成，包括年龄、性别、婚姻状态、出生地（美国某个州/地区或者其他国家）、职业、种族、学校出勤率以及地理区域。由于隐私约束，地理数据仅仅局限于粗粒度的区域如新英格兰和美国西海岸。结果数据的抽取是把520MB的Gzip文件解压缩为3.3GB的文本文件。

接下来，我不会赘述后续发生的详细细节。数据处理、清洗以及后续的导入操作是直接但繁琐的过程，最终生成了包含抽取出来的可查询形式的人口统计数据，保存在MySQL数据库中。为了帮助分析，我们使用星型模式来组织数据(http：//en.wikipedia.org/wiki/Star_schema)：把人口统计的评测方法存储到一张大的事实维表(fct table)，该维表的每一列表示一个人口统计学变量，通过键值表示变量的范围值。维度表集合存储了每个人口统计变量采用的键值的文本标签和描述。

这种组合方式提供了指导探索性分析的基础。我们通过发布查询生成数据摘要，这些查询通过选定的维度“向上钻取”(rlled up)数据。举个例子，我们可以通过对所有其他变量的样本人数进行求和，分离出年龄、性别和婚姻状态的关系。简而言之，我们拥有探索数据和原型系统可视化的基础。

可视化

给定人口统计数据的大小和范围，我们早就意识到把单一的可视化设计应用于所有的数据可能会导致某些灾难。只要有可能，我们就希望把数据转换成最简单的形式，用于支持广泛的分析。由于我们的设计需要面对广泛的受众，我们创建了一个可视化数据集合的方法，它表示选定数据的分片。在本质上，我们希望可视化有用，同时，它又尽可能地简单。

因此，我们的设计哲学需要理清哪一种数据集合可能是最大的兴趣点，而哪一种可视化设计和交互技术可以最佳支持对这些维度的积极探索。为了实现这一点，我们开始同时探索数据本身和可视化设计空间。

在设计一个界面来帮助别人探索数据之前，我希望首先能够确保数据本身足够有趣，使得别人愿意去探索它。我在探索中采用了很多种方法，包括SQL查询、Excel和可视化系统。数据探索最有用的工具是数据库可视化系统Tableau。使用Tableau工具，人们可以把数据库的各个字段通过拖拽方式映射到视觉编码中；然后，应用查询该数据库，对结果进行可视化（披露一下，我是Tableau软件的顾问）。因此，我们可以对很多不同方法构建可视化原型系统（见图12-3）。我们生成了一个庞大的数据可视化集合，和同事分享该集合，收集他们的反馈。使用真实数据快速评估可视化方法的能力为我们节省了很多实验时间，使得能够为最后的系统设计保留开发精力。

在探索数据过程中，我记录下了有趣的趋势、模式和发现的游离点。在某些情况下，有意思的故事是在很多维度的组合下发现的。例如，通过年龄和性别对婚姻状态的统计结果说明了女性初次结婚的平均年龄随着时间在增长，在过去一个世纪大约增长了5年。在其他情况下，根据单一数据类型描绘的曲线随着时间推移，揭示了很多有意思的故事，如农民工在劳动力市场中的盛衰以及全球不同的移民浪潮。一般来说，我发现对数据进行转换是很有用的，可任意把数据看做绝对人口数量，或者看做是十年内人口普查比例。

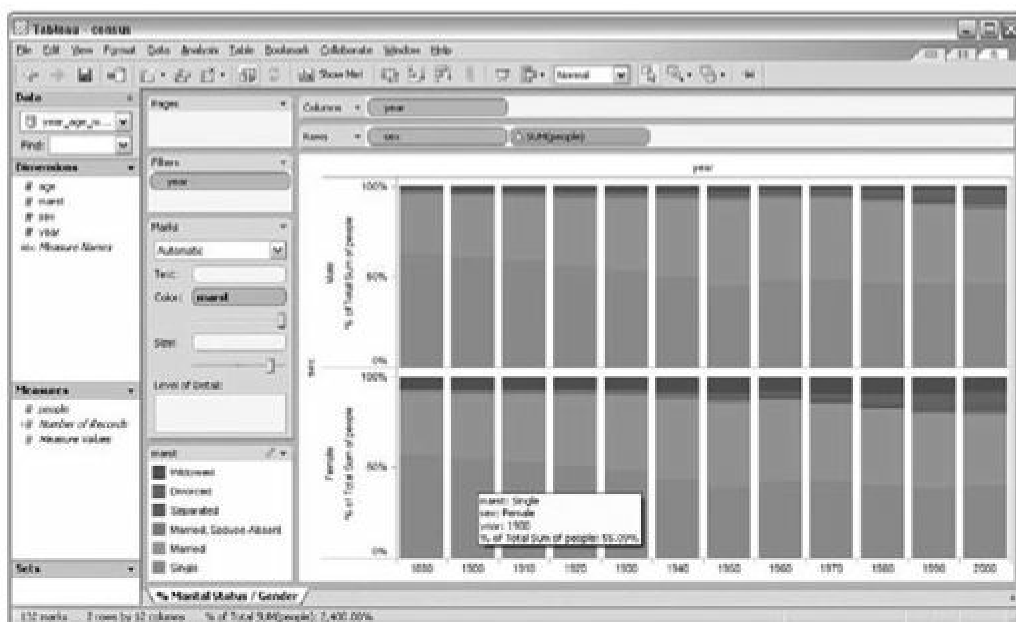


图　12-3：使用Tableau工具构建的显示几十年间婚姻状态分布的原型系统可视化（见彩图36）

设计考虑

有了原型系统，Martin、Fernanda和我为sense.us应用协作设计了交互式可视化。在设计中，我们首先简单列出了设计中的一些考虑。

培养个人相关性

如果没有人关心数据，就没有人会去探索它。我们假设熟悉的维度如地理和时间，使得用户能够快速地在数据和表单叙述中他们自己（或者和他们相似的人们）。假设我们可以访问的地理数据是有限的，我们集中于随着时间的变化来表示数据。我们还尝试使用交互技术，使用户可以快速找到他们感兴趣的数据记录，如特定的职业或国家起源。

提供有效的视觉编码

自然而然，我们希望使用视觉编码来帮助理解数据。在某些情况下，这种工作任务直截了当：如果想要检验两个数值之间的关联，我将很难找到比散点图更好的表示方式。在这种情况下，我们需要做出很多折中权衡。

对随着时间变化进行可视化的一个通用方法是使用线条图。但是，在一个线条图中显示200多个职业会导致封闭的线条之间非常混乱。取而代之，我们选择堆栈图(sacked graph)，它们可以把一个图叠加到另一个图上，从而可以对多条时间序列视觉上求和（见图12-4和图12-5）。我们的这种选择是受到Martin的婴儿名向导图(Bby Name Voyager)可视化的影响，婴儿名向导图是关于婴儿名字受欢迎程度的堆栈图，在网上变得超乎寻常地受欢迎(Wttenberg和Kriss 2006）。堆栈图可以清晰地显示聚集模式，并且可以很优雅地支持交互过滤，但是其代价模糊了个人倾向——对于某个倾向的理解是由栈序列的曲线带来的偏见生成的。相反地，我们确定点击一条序列可以过滤出对它的显示，因此，通过这种

方式，其趋势可以很容易被隔离出来进行显示。

进一步地，我们遵循了构建的文化约定，通过很多观察者所熟悉的方式对数据进行视觉编码（比如，用蓝色表示男孩，粉红色表示女孩）。当考虑如何在很多人口统计学变量间对交互进行可视化，而不是试图发明一些完全新颖的东西，我们采取的是对人口统计学家已经普遍使用的图形类型——人口金字塔进行扩展。

使每种展示方式都与众不同

在某些情况下，我们使用了相同的可视化类型来显示不同的数据类型。例如，对于职业和出生地数据，我们都使用了堆栈图。然而，我们希望每一种展示方式在视觉上都与众不同，这样用户就可以一眼识别出它们。结果是，我们为每一种可视化都构建了唯一的彩板。

支持直觉探索

为了培养交互探索，我们希望使得接口操作尽可能地简单。对于堆栈图，我们让用户输入搜索关键字对感兴趣的项进行查询。在其他情况下，我们提供了一组下拉菜单对维度进行选择或者过滤。我们还包含了控件来选择绝对人数计数和范化的比例。更高级的控制可能实现，而且我们发现提供该级别的控制能够促使通过整洁易用的接口所做的一系列探索。

积极参与玩耍

除了培养个人相关性，我们希望和该系统的交互有趣好玩。因此，我们致力于采用优雅高效的数据表示方式。我们设计了交互技术和动画

转换技术来提高响应和动态性效果，但是我们不想过度滥用这种文体特征。

我们希望增强而不是骚扰数据探索。我们不断改变动画形式和计时方式，直到设计上“感觉良好”。持续1秒钟的动画提供了不需要放慢分析的过程，观看者就可以跟上的视觉转换过程。

可视化设计

首先通过参与自己的数据探索，我们可以确定自己最感兴趣的数据维度。为了设计一组人口统计数据的可视化解决方案，我们把这些观察应用到设计考虑中。

工作向导

“工作向导”(Jb Voyager)是一个堆栈图，显示了美国在过去150年间（见图12-4）的劳动力的组成部分。每条序列曲线表示一种职业，通过性别进一步细分：蓝色代表男性，粉色代表女性。用户可以通过点击一条序列曲线，只显示其相应的职位，或者通过输入关键字查询来过滤掉与查询不匹配的职位。我们还包含了下拉菜单，通过性别进行过滤；在绝对人数计数和劳动力比例之间进行视图转换。这些操作支持对聚集趋势（如第二次世界大战后涌入劳动力市场的女性）和个人模式（如机动车工程师的盛衰）的探索。

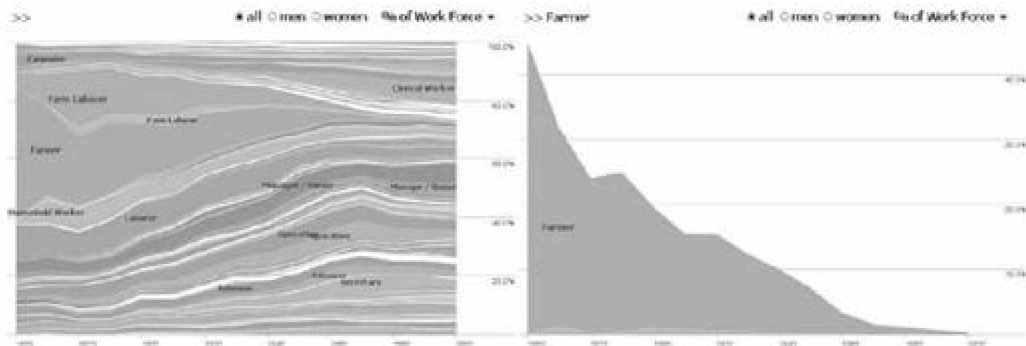


图　12-4：工作向导可视化：左图显示在过去150年间劳动力组成概要视图，右图显示过滤后农民工的比例的视图（见彩图37）

我们很快意识到仅仅通过性别对序列曲线进行着色是不够的。当我们过滤了视图，仅仅显示男性或者女性，要区分不同的序列曲线变得很困难。一个解决方案是通过任意方式改变色彩饱和度，使得感官上能够区分不同的序列曲线。Martin提出了一种很有技巧的变换方式：不是随意改变颜色，而是通过有意义的、数据驱动的方式来实现。因此，我们根据每个职业的社会经济指数分值来改变其对应的颜色饱和度。因此，平均收入越高的序列曲线，其颜色越深。在实践中，这种编码方式工作良好，不需要为显示增加误导或者无意义的视觉特征，而提高了不同职业的识别度。

出生地向导

“出生地向导”(Brthplace Voyager)和工作向导的设计相似，但是它显示的是每个人口统计年份的美国居民的出生地（见图12-5）。其记录的出生地或者是美国某个州和地区或者是其他国家。交互式控件可以通过查询关键字或不同大陆位置进行过滤，只显示绝对计数和比率。

数据可视化支持对全世界（如过去来自欧洲的移民浪潮以及当前来

自拉丁美洲的移民浪潮）和美国国内（如每个州的居民的变化比率）的移民趋势的探索。

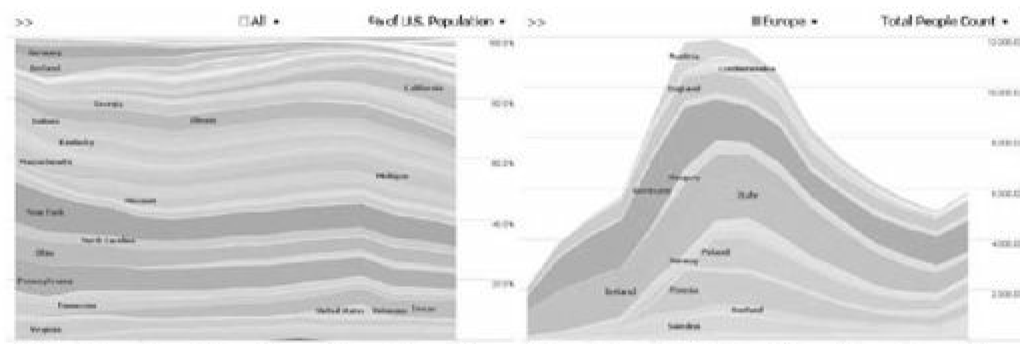


图　12-5：出生地向导可视化：左图显示在过去150年间出生地的分布概要，右图显示过滤后欧洲移民的总数的视图（见彩图38）

在出生地向导中，我们遇到了和之前相似的着色问题。在这种情况下，我们根据大陆位置来分配色彩，为美国的各个州增添一些额外的色彩。然后，我们实验了改变色彩饱和度的不同方法，直到我们可以使用在该州出生的人口总数或者所有时间片断的国家作为后备数据。

美国人口统计的国家地图和散点图

虽然工作向导和出生地向导的时间线在设计上都是使观察者以历史叙述的方式进行参与，但是我们希望能够包含更加传统的视图。我们提供了一个着色的美国国家地图，显示每个州的人口统计变量的分布。在2000～2005年（如图12-6的左图所示），在国家人口变化的标注地图中，人们可以看到西南部地区人口得到显著增长，而North Dakota的人口却在不断下降。我们还提供了散点图的表示方式来检测不同变量之间的潜在关联。用户可以把人口统计变量映射到坐标轴的x坐标和y坐标，圆圈的面积表示整个美国。例如，人们可能会注意到全国家庭收入和零

售额之间的关联（如图12-6的右图所示）。这些视图的备份数据包括我们从美国人口统计局网站上下载的额外的统计数据。

人口金字塔

最复杂的可视化是交互式的人口金字塔，它是为了同时帮助探索多个人口统计学变量而设计的（见图12-7）。人口金字塔（有时称之为“年龄-性别”金字塔），是由同事介绍的关于人口统计学的一种图形。金字塔被一条垂直曲线划分为两半：一条曲线表示男性，另一条表示女性。y轴表示年龄，通常以5年或10年作为一组进行分组；x轴表示某个年龄段的总人数，男性的人数值朝一个方向增长，而女性的人数值朝相反方向增长（如图12-7的左图所示）。该金字塔形状传递了人口动态信息：险峻的金字塔暗示了其死亡率比圆柱形的更高。

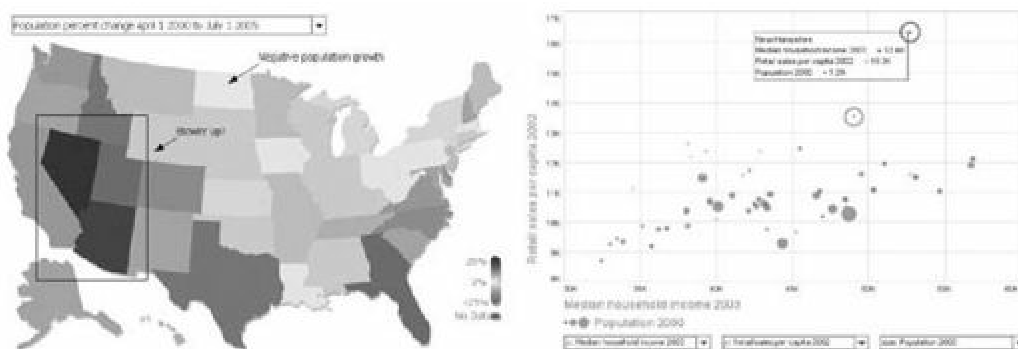


图 12-6：（左图）交互式国家地图显示每个州从2000～2005年的人口变化，（右图）美国散点图显示了平均家庭收入（x轴）和零售额（y轴），新罕布什尔州和特拉华州有最高的零售额（见彩图39）

我们创建了一个交互式金字塔，除了年龄和性别，它还结合以下人口统计变量：地理区域、种族、婚姻状态、学校出席率和收入水平。该金字塔的两边默认通过性别来划分数据。我们放宽了这种限制，增加了

下拉菜单，通过该菜单，用户可以选择一个人口统计学变量，把该变量对应的（x,y）两个值映射到金字塔的两条边上。比如，用户通过把美国西部海岸放到金字塔一边，而把新英格兰放在另一边的方式来观察地理区域。

我们还引入了彩图编码菜单：选择一个人口统计变量把金字塔的两条边转换为描述变量值的流行率(pevalence)的堆栈图。例如，用户可以以学校出勤率进行统计，用于查看学校人口出勤率分段情况（如图12-7的右图所示）。我们为每个变量选择一个与众不同的彩板，只要可能就依赖于已有的文化约定（如蓝色=男性，粉色=女性），使用在线工具ColorBrewer(http：//colorbrewer.org)来确定其他情况的颜色选择。我们使用灰色带表示数据丢失或者未知值。

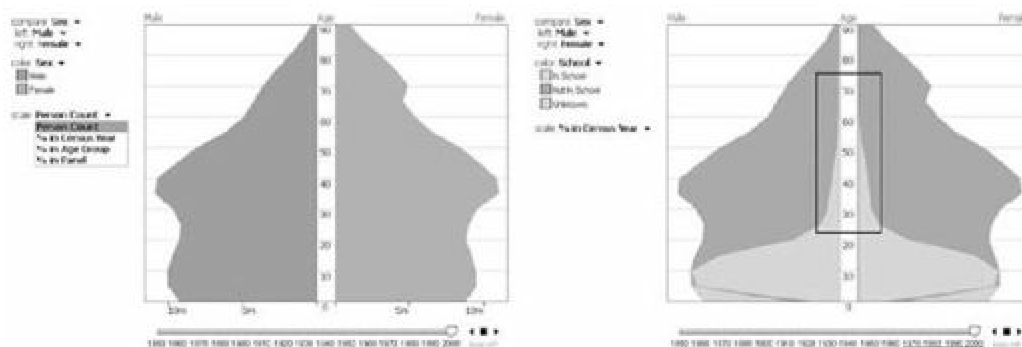


图 12-7：人口金字塔可视化：（左图）2000年的每个年龄组的男性和女性的总人口数量的比较，（右图）2000年学校出席人数的分布（标注重点突出了成人教育的流行率）（见彩图40）金字塔下方的时间线使得可以在整个人口统计期间进行时间探索，而回放特征使金字塔可以随时间动态变化。举个例子，随着时间对人口变化的动画显示显著突出了战后期间

社会出现的婴儿高潮。由于多层次色彩和泡沫式动画的结合，用户亲切地把我们的人口金字塔重命名为“Georgia O'Keefe的熔岩灯”(lva lamp)。

最后，我们支持四种尺度来衡量数据：人口总数、10年内人口比率、（金字塔）图面内(pnel)比率（当金字塔两边不协调时有用）和年龄组比率（为了探索不同年龄间的比率区别）。我们发现每一种衡量尺度都有助于揭示某些特定的故事。例如，年龄组比率显示说明了年纪大的男人比年纪大的女人更容易结婚，推测原因是女人平均活得更长，因而成为了寡妇。

实现细节

我们把每一种可视化作为Java Applet(Java应用小程序）实现，这样我们可以把它嵌入到Web页面。选择Java而不是Flash，一部分是由于性能原因，但更重要的是当时有现成的Java可视化框架。堆栈图和人口金字塔是通过开源工具包Prefuse(http：//prefuse.org)构建的。每一种可视化是通过从人口统计数据库中抽取出来的纯文本文件来支持的。对于人口金字塔，我们为每种可能的人口统计变量组合生成了一个文件，预计算数据的所有相关投影。这种方式避免了在服务器端处理数据的需求，生成方便管理的存储历史：虽然刚开始数据库包含3GB多的数据，在最后部署时数据减少到仅仅稍多于3MB！当然，这种方法确实存在其局限性：它阻碍了用户探索新的人口统计变量组合，并且使得要求服务器端处理数据的可视化变得复杂。

协作

因而，我们创建了sense.us网站，它通过协作分析机制对可视化进行耦合（见图12-8）。左图是可视化Applet（见图12-8中的a）和标注工具（见图12-8中的b）。右图提供了图形化的书签线索（见图12-8中的c），提供了用户保存的访问路径以及显示和当前视图相关评论的讨论区域（见图12-8中的d和e）。我们通过一组协作特征来增强可视化，这一点在后面会详细描述：视图分享、双向链接讨论、图形化标注、书签线索和以评论列表和用户行为个人档案生成的社会导航。

视图分享

当围绕可视化协作时，我们认为参与者看到的必须是相同的视觉环境，才能够为其他人的行为和评论打下基础。为此，sense.us提供了对视图添加书签的机制。我们试着通过把应用绑定到传统的Web书签上，使它透明化。浏览器地址栏总是显示链接到可视化的当前状态的一条URL，该URL由一组过滤、导航和视觉编码等参数进行定义。随着可视化视图的变化，URL也随之更新来显示当前状态（见图12-8中的f）；用户可以在任意时刻剪切、粘贴一条链接到当前视图，因而简化了通过Email、博客或者即时通信来共享视图的过程。为了遵循用户期望，把浏览器的前进和后退按钮绑定到可视化状态，这种方式可以很方便地导航到过去浏览的视图。

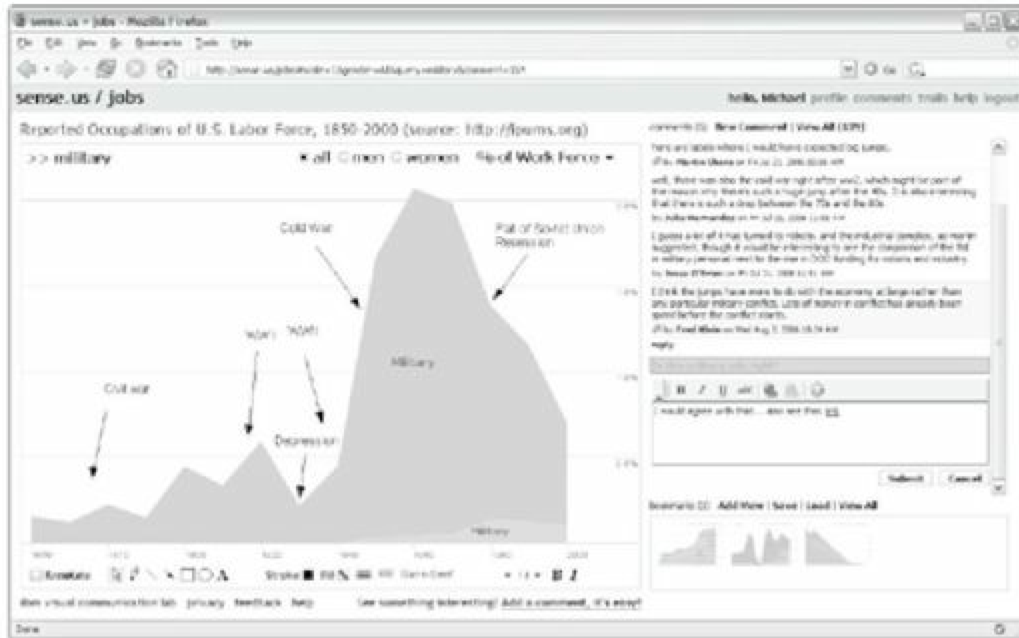


图 12-8：sense.us协作式可视化系统：a）交互式可视化应用小程序，图形化标注出当前选定的评论。该可视化是按照性别划分的美国劳动力人数的叠加式时间序列可视化。该图显示了军事职位的劳动力比率；b）一组图形化标注工具；c）保存的视图书签索引；d）添加评论的文本入口区域，可以拖动书签到文本区域，为评论中的视图增加链接；e）在当前视图添加相同主题的评论；f）该应用当前状态对应的URL。该URL随着可视化状态变化自动更新（见彩图41）

双向链接讨论

为了设置可视化相关话题，我们创建了称之为“双向链接讨论”(dublylinked discussion)的技术。该技术起始于一个独立的讨论页面，在这里用户可以为某种特定的可视化状态（或视图）添加评论。评论显示在Web页面的右侧，通过线性的讨论话题进行组合（见图12-8中的e）。每条评论显示了话题主题、评论文本、作者全名以及评论时

间。点击一条评论可以把可视化切换到书签状态，表示该评论员已看过的视图。

用户可以通过开始一个新的话题或者回复已有的话题来增加评论。当点击“新的评论”或“回复”链接，该站点就会显示一个文本编辑器，可以通过它添加评论（见图12-8中的d），图形化标注工具（后面讨论）也变成可操作模式。在提交时，评论文本和任何标注都会被发送给服务器，并且评论列表会被更新。

以上描述的界面基于把评论生成可视化的链接。系统还提供了另一个方向的链接：从可视化到讨论。因为用户可以改变可视化中的参数和视图，他们可能会无意中改变了另一个用户已经添加评论的视图。发生这种情况时，相关的评论会自动出现在右侧面板。我们的直觉是这种“双向链接”讨论界面结合了独立的和嵌套的讨论的各个方面，有助于为可视化奠定基础，并促使可视化本身变成一种社交场所。

我们很快意识到书签机制不足以支持双向链接讨论。为了观察从一种可视状态链接到该视图的所有评论所面临的挑战，让我们以图12-8的可视化为例。当用户在最上层搜索框输入“military”（军事）（见图12-8中的f），他看到所有以“military”字符串开头的职位标题。而该用户如果只输入“mili”，他看到的是所有以“mili”开头的标题——但是其结果是和之前完全相同的职位信息。不同的参数设置生成不同的URL结果，但是提供完全相同的可视化视图。从更广义上说，参数设置可能不包含一对一的可视化状态映射。因此，为了对视图添加评论，我们需要一种索

引机制，它可以识别相同的可视化状态，虽然其参数表现形式不同。

我们通过区分两种类型的参数来解决索引问题：过滤参数和视图参数。过滤参数确定哪些数据元素在显示中可见。我们不是直接索引过滤参数，而是通过记录哪些数据项当前可见来索引应用的过滤状态，因此可以捕获不同的过滤参数来最终生成相同的过滤状态的情况。而视图参数则是调整视觉映射方式，如选择范化的或绝对的坐标轴尺度。我们当前的系统直接索引了视图参数。书签机制同时基于过滤状态和视图参数，通过计算概率唯一的SHA-1散列值实现这两种参数的索引。通过将散列值作为键值，即可检索当前可视化状态的评论。

通过图形化标注来指向标识

在物理协作中，人们通常通过言谈举止，尤其是指向某个物体来表示这个物体，交谈该话题。在分布式、异步Web上下文中，图形化标注可以发挥相似的通信作用。我们假设图形化标注对于指向行为和有趣的评论都将非常重要。为了给评论增加图形化元素或者指向感兴趣的某个特征，用户可以使用绘图工具（见图12-8中的b）来标注评论的视图。这些工具允许在可视化视图中绘制自由格式的笔墨、线条、形状和文本。这些工具和那些表示工具如Microsoft PowerPoint相似，目的是充分利用用户对于系统的熟练程度。包含标注的评论是通过在评论列表中作者名字的左侧显示一个小图标来表示（见图12-8中的e）。当鼠标悬停在一条标注的评论上时，该评论将以黄色突出显示，并呈手形光标显示。因此，点击包含标注的评论区域会显示标注，页面亮度变暗，使得

评论区域变得透明。再次点击评论（或者点击不同的评论）将会删除当前的标注和高亮。

图形化标注的形式是在可视化上绘制矢量图形。当提交一条新的评论时，浏览器从可视化应用小程序请求当前标注（如果有的话）。该标注是以XML格式保存，然后通过gzip压缩，以Base64字符串表示方式编码，然后传给浏览器。如果后期从服务器检索到评论，编码过的标注就以JavaScript变量的形式保存在浏览器中。当用户请求显示某条标注，该编码过的标注就会传递给应用小程序，然后应用小程序对其解码并绘图。

我们称这种方法为几何学标注，类似于在可视化表现方式上覆盖了一层“乙酸纤维层”，它和数据相关的标注不同，后者直接和底层数据关联。我们选择实现自由格式的标注机制，因此，我们可以在无约束方式下，首先研究指向行为。这种方式除了为我们提供自由表达方式，几何学标注还包含技术优势：它允许对整个可视化重用相同的标注系统，简化实现并维持一致的用户体验。

收集和链接视图

在数据分析中，通过观察数据，对不同方式进行比较是很常见的。而且，（看图）讲故事已经在可视化的社交应用中发挥了重要的作用。绘画比较和讲述故事都需要能够把多个视图书签嵌入到一个评论中。

为了支持这种多视图评论和叙述方式，我们创建了“书签索引”(bokmark trail)向导小工具。书签索引在功能上类似购物车：用户通

过导航访问站点，她可以点击专门的“添加视图”链接，把当前视图添加到书签的图形列表（见图12-8中的c）。任意个数的可视化书签都可以增加到一条索引书签中。可以命名并保存一条书签线索，使得其他用户可以访问。

当做出包含多个视图的链接的评论时，书签线索向导小工具还可以作为短期的存储机制。从书签线索中拖动一个缩略图，把它放入一个文本区域，为书签视图创建一条超链接；然后，用户可以直接在文本编辑器中编辑或删除链接文本。当鼠标悬浮在链接文本上时，显示该链接视图的技巧提示小图标。

意识和社会导航

术语“社会导航”(Scial Navigation)指的是基于别人的行为或建议进行导航的倾向。在Web上，为了提供额外的导航选项，这种导航方式可以通过显示其他用户的使用历史来实现。我们设计了sense.us应用来支持通过评论列表和显示用户最近行为的个人档案页面的社会导航。评论列表提供了该系统用户做出的所有评论的一个可搜索集合，可以对该列表过滤，从而集中于单方面的可视化（见图12-9）。评论列表页面包括文本和每个评论的可视化状态的小图标头像。悬停在小图标上可以生成技巧小提示，图像也变大了。点击评论链接会把用户带到该评论的可视化状态，显示该评论中包含的任意标注。发表评论的作者名字链接到他的个人信息页面，该页面包括他的五个最新评论话题，以及五条最新保存的书签线索。这种视图还标识了从用户发表的上一条评论来开始，该话

题的评论数量，这种方式允许用户监测他们所参与讨论的话题的相关动态。

虽然可以有更复杂细致的社会导航机制，但是我们希望仅仅通过这些基本的选项来观察系统的使用方式。我们对观察数据驱动的探索和社会导航之间的相互影响特别感兴趣。当用户探索数据时悄悄检索讨论，可以为探索过程引入潜在可能相关的谈话。同时，评论列表和最近话题动态可能有助于用户找到感兴趣的视图，使得社交活动成为数据探索的催化剂。

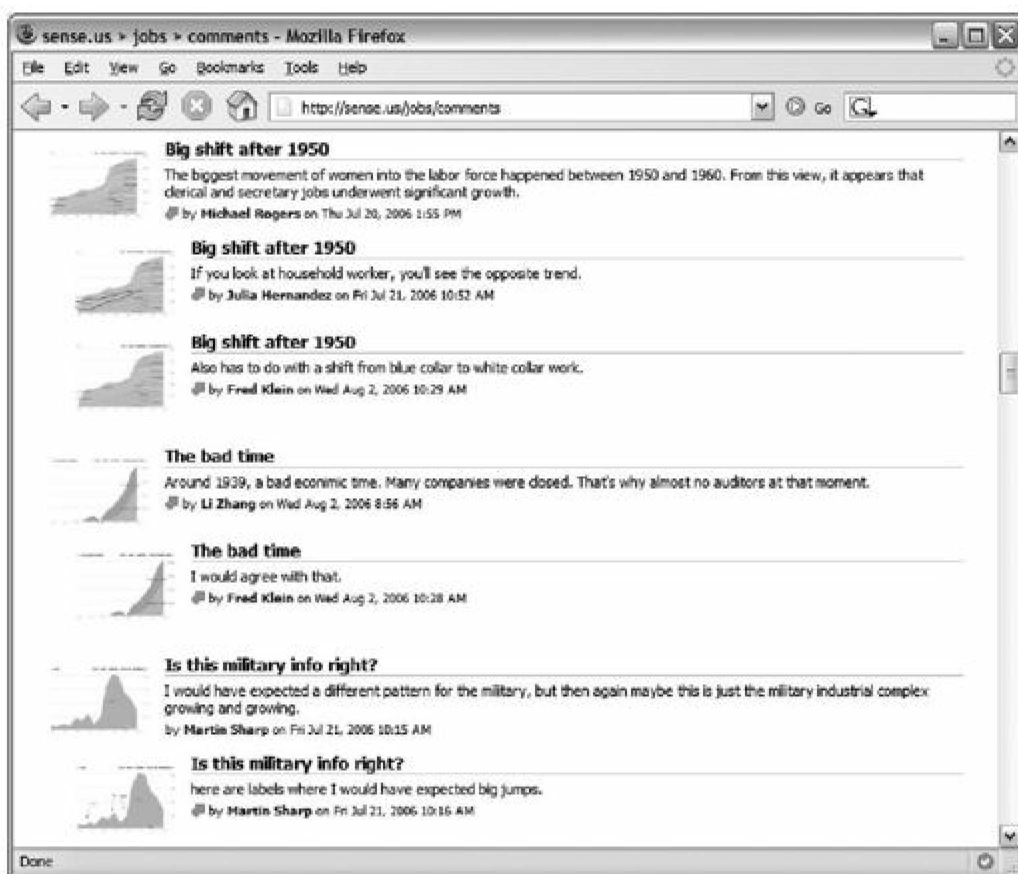


图　12-9：sense.us评论列表页面，评论列表显示可视化的所有评论，提供链到评论的可视化视图的链接（见彩图42）

不唐突的协作

我们还遵从计算机支持的协作领域的通用设计指南：协作式特征不应该妨碍个人使用。因此，我们默认上并没有给出视图标注。相反地，可视化的评论显示在屏幕右侧不显眼的地方，而图形化标注则是根据用户需求，“随需显示”。

“向导”和“偷窥”

在完成数据获取、设计和系统实现步骤后，我们的网站就可以运行，准备好让用户“现场试验”。本着一组用户研究的想法，我们部署了系统，观察人们对系统会做出什么反应，产生什么样的见解，以及我们应该如何改进改站点。

我们邀请了30名用户到我们的实验室，用来观察他们如何使用sense.us探索数据。每个人可以看到其他参与者对该站点的贡献。我们还在IBM企业内网现成部署了sense.us，这样公司内所有员工都可以访问。从这些研究中，我们调查了人们如何参与可视化，以及协作特征如何给人们的探索带来影响。下一步，我将总结观察到的一些最有意思的使用模式。

寻找模式

绝大多数用户的第一反应是参与“寻宝游戏”(savenger hunts)，获取有趣好玩的观察，通常是由个人所处环境所驱动。例如，用户会搜索他们自己或朋友或家庭成员的工作，或者查看他们祖先的出生地数据。在探索过程中，人们通常会做出评论，记录他们认为最有意思的趋势。

举个例子，参与者注意到在1930年酒吧服务员的人数减少为零，发表评论说其原因是由于禁酒令。有人发现在19世纪从加拿大移民的（美国）人口比率上升到最高值，然后稳步下降，就咨询是什么原因导致了这种趋势。而另一个用户注意到在大萧条后股票经纪人数骤减，留下了

如图12-10所示的视觉评论。

用户还通过人口金字塔发现了有趣的趋势。例如，用户探索了婚姻状态随时间的变化（见图12-11）。绿色带和紫色带意味着分离和离婚的流行率，该比率在1960年后显著上升。一名用户调研了学校出席率，发表评论说从1960年后，成人教育显著增长（如图12-7的右图所示）。

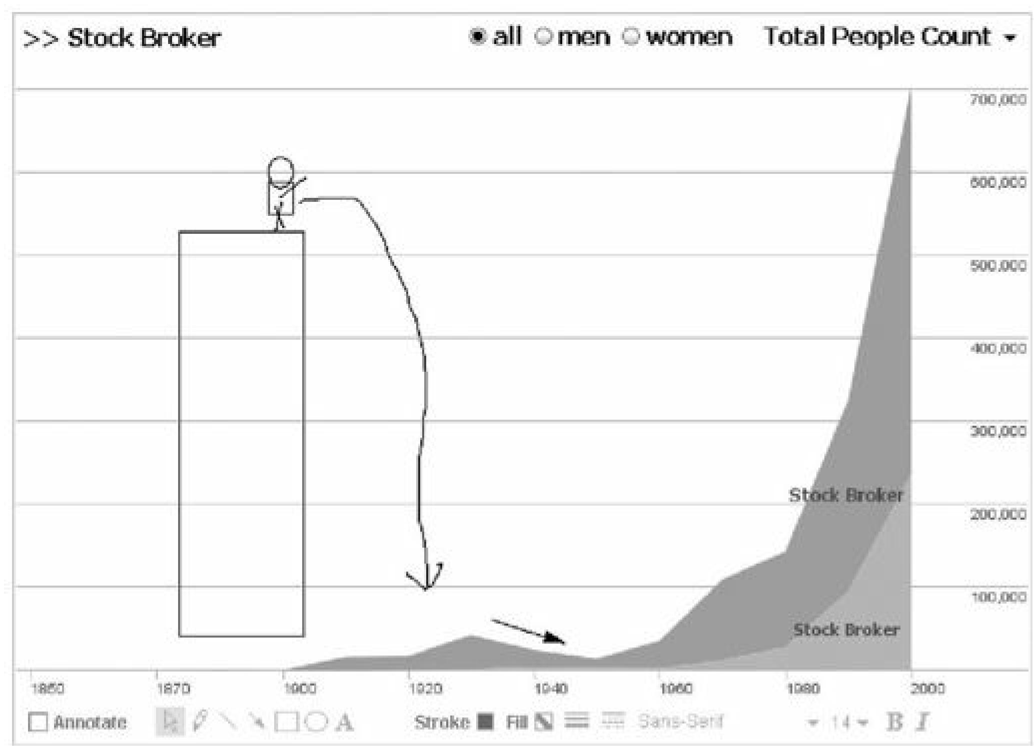


图　12-10：股票经纪人的标注视图，其评论意思是“大萧条‘杀死’了很多经纪人”（见彩图43）

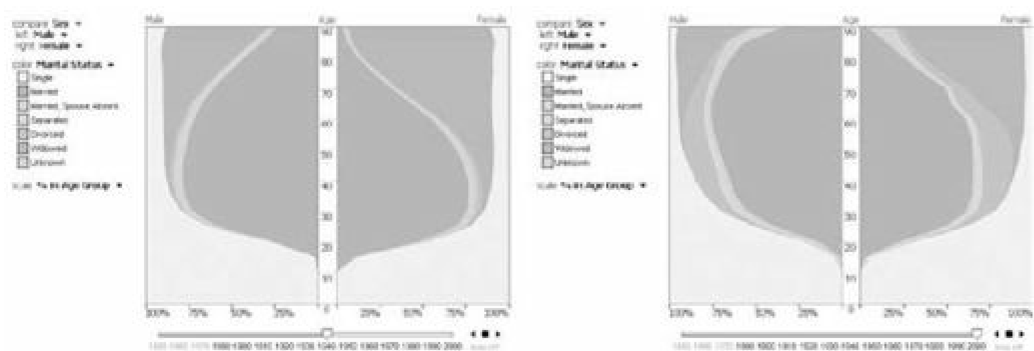


图　12-11：人口金字塔显示在1940年（左图）和2000年（右图）每个年龄组的婚姻状态分布（见彩图44）

另一种情况是，一个用户把金字塔的两边映射到大西洋中部地区（即纽约、宾夕法尼亚州和新泽西州）和西海岸地区的人口（见图12-12）。在1850年，“淘金热”年代，西海岸地区的人口和东海岸地区有很明显的区别，前者主要是年青和中年男性。90年之后，两个地区的人口统计则更接近对齐，虽然有用户提出西海岸地区人口在比统计的晚10年后才开始平滑。

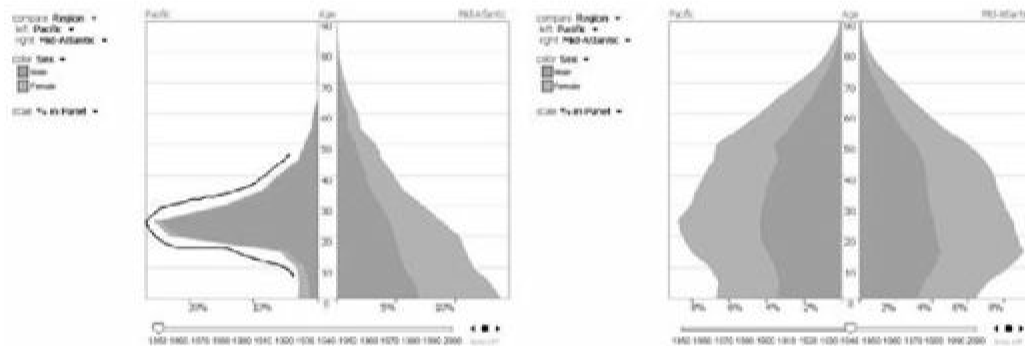


图　12-12：人口金字塔比较了在1850年（左图）和1940年（右图）西海岸和大西洋中部地区的人口（见彩图45）

比起特定的视图，有些用户则对重现模式更感兴趣。一个用户对探索历史上男性主导职场的职业生涯感兴趣，但是发现在后半个世纪，女性的人数不断增加。用户系统性地探索数据，把视图保存在书签线索中，然后命名为“女性的崛起”，并和其他人分享。

类似地，一个更有数学思想的用户对于工作波动模式感兴趣，他创建了一条曲线，显示工作波动重现概率。另一个用户搜索被科技所取代的职位，如银行出纳员和电话操作员。每一种情况，其结果都是通过多

视图所“蜿蜒”出来的一次旅行或者一个故事。

使一切变得有意义

用户观察了数据，他们通常会张贴问题寻找解释，或者假设该数据可能导致产生某种趋势。很多这种问题和假设吸引了其他用户的回复，开始了对社会理解的一个循环过程。在我们的现场部署中，一个用户对散点图评论咨询为什么新罕布什尔州有这么高的人均零售额（见图12-6）。另一个用户指出新罕布什尔州没有征零售税，而拥有美国第二大零售业的特拉华州也没有征零售税。照这种方式，讨论通常涉及引入可视化中不包含的上下文信息。比如用户迭代式构建了一条事件时间线来标注军事建设（见图12-8），而另一个用户标注了引入义务教育的教师图。

社交数据分析的一个例子是牙医在劳动力比率中的盛衰以及复苏（图12-13）。第一个评论指出了这种趋势，然后咨询其发生的原因。一条作为独立的话题回复该问题的主题为“和氟化作用有关？所以上升……和溺爱小孩，给他们吃了很多糖有关？”；而另一条回复主题是“由于预防性的牙医行业变得更有效，牙医寻找继续工作的方式（如很多人现在每年看两次牙，而十几年前每年只看一次）”。然而，最有说服力的，可能是包含链接到不同观点的评论，同时显示牙医和牙科技术。因为牙医人数比例已经下降了，而牙医人数又得到了显著增长，意味着在该领域内的专业性。针对这条评论，另一个用户问：“我想知道如果学校学牙医的费用变得太昂贵以至于人们根本不敢想，或者当他们

上技校后，牙医从业费用很高，那牙医数量会减少吗？”视觉数据分析、历史知识和个人轶事都对理解过程(snsemaking process)有影响，阐明了塑造该数据的各种因素。

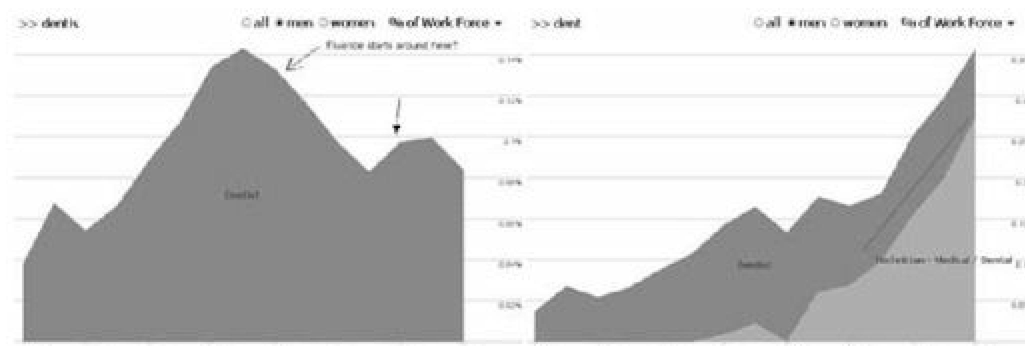


图　12-13：标注的工作向导视图（左图）突出显示了1930年后牙医人数的减少，（右图）由于牙医人员排序的上升，牙医从业人数得到增长（见彩图46）

评论的另一个作用是帮助解释数据，尤其对于数据集合中模糊的涵义或异常现象。对于IPUMS项目，虽然我们很努力工作，但是丢失数据以及标签晦涩的问题依然存在。为了比较所有人口统计普查年份的数据，必须形成共享的分类机制。以职位数据为例，使用的是20世纪50年代的模式。该模式不包含一些现代社会的职位如计算机程序员，而且一些标签含义不明。

一个很显著的职业被标记为“技工”(Oerative)，泛指主要包含有技术的劳动力。这个术语对于用户几乎没有意义，其中一个用户问：“技工到底是啥？”其他用户回复同样不解或者提出某种理解，如“我肯定它指的是工厂工人”。另一个人同意，指出工人的数量很大，技工的盛衰时间似乎和工厂的机器操作员的盛衰一致。

照这种方式，用户集体参与数据验证和解疑中，通常在数据中“种”下了“标记贴”(sgnposts)来帮助其他人理解。总的来说，大约16%的评论涉及数据命名、分类和收集问题。

人群中“冲浪”

我们观察到绝大多数最初是由他们自己的兴趣或者在概述中找到的兴趣项所驱动来探索数据的（如“噢！看看这些可怜的农民是怎么消亡的”）。最后，用户将没有更多的想法或者厌倦了探索。这时候，我们观察的每个用户都是不再探索可视化，而是去探索评论列表。有些用户认为这么做他们可以更快地发现有意思的观点。表达这种意思的评论包括：“我相信其他人发现了一些更有趣的东西”和“我需要站在别人的肩膀上”。

其他试验者对于他们知道的某些人或者发现其他人所调查的东西感兴趣。一个用户说：“我觉得自己就像数据偷窥者。我真的很喜欢看别人都在搜索什么。”在数据驱动的探索和社会导航之间切换很常见：通过评论列表发现的视图通常可以点燃新的兴趣点，并激励更多的可视化过程中的数据分析。在经过一些探索之后，参与者习惯性地回到列表寻求更多的兴趣灵感。因此，我们观察到在数据驱动的探索和社会导航之间正面的反馈循环：探寻社交活动有助于促进探索新的分析问题。换句话说，用户可以很流畅地在“向导”和“偷窥”两个角色之间切换。

结论

基于sense.us项目成果，我们发现交互式可视化和对社会的理解可以帮助用户更丰富地探索数据集。然而，作为研究原型系统，sense.us网站从未对外公布过。相反地，我在IBM的同事基于sense.us，发布了Many-Eyes.com站点：一个公共网站，用户可以上传他们自己的数据集，使用各种交互式可视化组件对数据可视化，参与在线讨论或者在该站点外部的博客和wiki中嵌入可视化视图。

本着同样的精神，Web服务如Swivel.com和Data360.org，以及商业产品如Spotfire Decision Site Posters和Tableau Server，都允许用户在Web上发布可视化帖子，吸引其他人参与社交数据分析过程。和较大的Web规模的社交计算的运动同时，仍然存在如何促进和支持社交形式的数据探索。关于如何集成数据分析和社交活动的很多有趣的研究问题仍然亟待解决。开发问题包括设计更好的社会导航提示、更丰富的标注技巧和结合用户的观察、问题和假设到推理分析故事中的新方法。

虽然我们在sense.us网站上看到的分析形式本质上是解释性的，但是该系统有明显的教育价值，而且用户反馈说使用sense.us既让人开心又能够获取信息。此外，用户生成的很多观察、问题和假设引起了专业分析师的跟进分析。数据可访问表示与社会交互相结合，有助于人们把数据转换为对社会更丰富的理解。

我发现这一切真的很美！

参考文献

[1]Anscombe,Francis J.（1973）."Graphs in Statistical Analysis."American Statistician，27，17-21.

[2]Bertin,Jacques.（1967）. Sémiologie Graphique,Gauthier-Villars.English translation by W.J Berg as Semiology of Graphics,University of Wisconsin Press，1983.

[3]Card,Stuart K.， Ben Shneiderman,and Jock D. Mackinlay.（1999）.Readings in Information Visualization:Using Vision To Think,Morgan-Kaufmann.

[4]Cleveland,William S. and Robert McGill.（1985）."Graphical Perception and Graphical Methods for Analyzing Scientific Data."Science，229（4716），828-833.

[5]Croes,Marnix.（2006）."The Holocaust in the Netherlands and the Rate of Jewish Survival."Holocaust and Genocide Studies，20（3），474-499.

[6]Robison,Wade,Roger Boisjoly,David Hoeker,and Stefan Young.（2002）."Representation and Misrepresentation:Tufte and the Morton Thiokol Engineers on the Challenger."Science and Engineering Ethics，8，59-81.

[7]Tufte,Edward R.（1997）. Visual Explanations:Images and

Quantities,Evidence and Narrative,Graphics Press.

[8]Ware,Colin.（2004）. Information Visualization:Perception for Design,Second Edition,Morgan-Kaufmann.

[9]Wattenberg,Martin and Jesse Kriss.（2006）."Designing for Social Data Analysis."IEEE Transactions on Visualization and Computer Graphics，12（4），549-557.

第13章　数据所做不到的

Coco Krumme

数据做了很多事情：它使我们能够区分科学与迷信，重复与随机。在过去几十年中，科学家们收集、整理和存储数据的能力已经有了长足的发展。从医学的决策制定到软饮料的市场营销到供应链的管理，我们更多地是依靠事实而不是直觉。我们依靠数据驱动这一切。

但是，数据无法驱动一切。在刚刚过去的20世纪，心理学家对人们对任何事物的解释接近理性式的稳重这一理论已经“颇有微词”。实际上，我们在解释信息的时候通常是有倾向性的。此外，真实世界并不是和骰子游戏的概率一样简单。相反地，个人必须从观察到的经验模式中提取可能性。

本章是关于存在于数据和分析之间的近似和偏见的讨论。它描述了一组新的实验和工具来帮助我们更有效地利用数据，而且从医学、消费和金融决策领域不断增加的文献中抽取一些例子。

假设我请你从图13-1中识别出与众不同的一只鸭子。

答案很简单且可以立即给出：人类的眼睛可以从一群白天鹅中识别出丑小鸭。但是，对于计算机，其解决方案就没就有这么简单：对于绝大多数情况，图像识别软件远落后于人类的视觉系统。

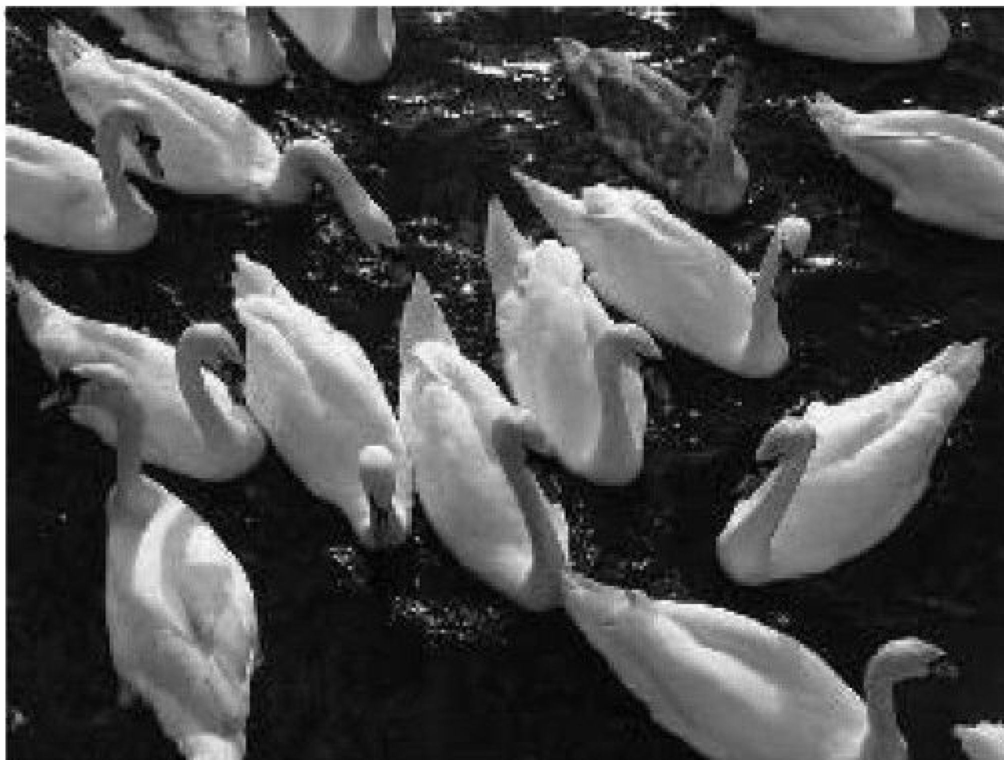


图　13-1：人类视觉系统善于识别出与众不同的鸭子

我们的大脑能够快速地对新事物和旧事物进行对比。我们善于进行类比和扩展：将手套、毛衣和滑雪者关联在一起；卡通老鼠和卡通熊是相似的；卡通熊和真正的熊相似；后两类的相似性本身很不一样。

即使你抹去上下文，我们的视觉系统仍然非常擅长挑选模式。以图13-2为例：你将如何把这些数据分成两组？

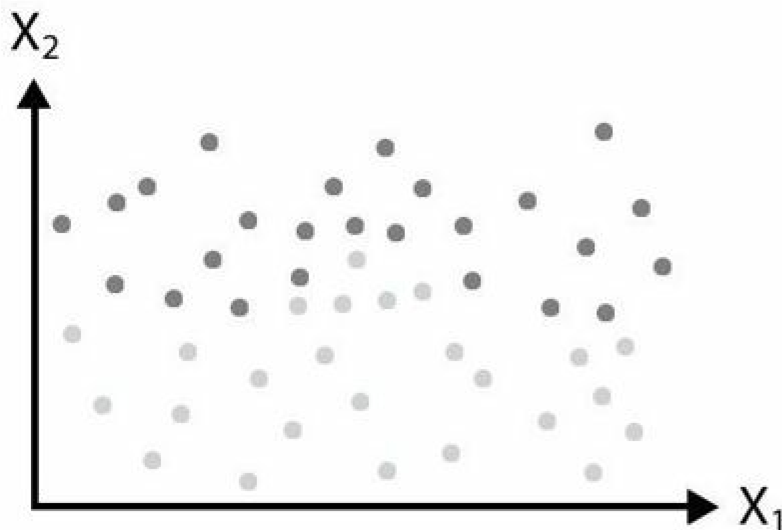


图　13-2：我们可以构建模型来区分两组数据集（见彩图47）

从添加维度和数据点开始，该任务如果由人来做，将会逐渐变得太复杂；而计算机则可以较轻松地解析该数据。你或许可以在两个点集之间画出一条近似曲线，但是计算机能够构建一个模型进而以最精确的方式划分数据。

现在，看看图13-3的图表模式，它显示了三个主要制造商在2005年1～10月这段时间的股票价格。观察以下图表，在阅读下一段前给出答案：基于给定的模式，你预期这三只股票中哪一只股票在2006年会涨？

你是选择了第三只股票吗？因为它看起来最后在涨。那么，你错了。事实上，如果你选择第一只或者第二只股票，你也一样错了。没有哪只股票会上涨，也没有哪只会下跌。实际上，这三个图表甚至并不能表示股票价格。它们是随机生成的：垃圾数据——可能有人会如是说。但是一旦相信这些图表属于某个公司（在本例中指制造业），就会产生

各种各样的猜测。即使有真正的图表，除非你对某个公司或者行业有专业知识，否则很难基于一只股票过去10个月的业绩预测它将来的走势。



图 13-3：三只股票（a、b和c）在2005年的业绩（见彩图48）

从噪音数据捏造故事的倾向有时被称为“叙述谬误”(nrrative fallacy)。即使你对问题持怀疑态度（或者读得太快），当我们调查一些拔尖的MBA学生——其中有些应聘金融业的工作——相同的问题，他们很确定地表达对这些“股票”走势的信心。有些人说会涨，有些人说会跌。当为这些图表补充一些随机生成的“新闻剪辑”，并随机和某个图表放在一起时，这些学生更坚定了自己之前的预测——这些可能显示了人们根据数据给自己讲一个好故事的能力(Kumme，待发表)。（再看一下财经

记者写的一些俏皮话：“对失业率上升的恐惧导致道琼斯指数下跌100点。”）

如果说人类善于观察模式，则人类同时也是编造关于统计学故事的大师。如果我们知道数据来自哪里及其涵义，这个问题就不那么严重；如果我们面对的是来自很多数据源的证据和高赌注的产出时，这个问题便可能会是灾难性的。

最后一个例子，在探讨问题之前，想一想你的期望。把过去的结论应用于当前分析的倾向被称为“确认偏见”(cnfirmation bias)(Lrd等1979）。（想一想你认识的某些人，他们只是为了选出一些语句来进一步肯定自己的世界观而阅读。）这在处理数据时也是真实存在的现象。观察表明，科学家更偏向于坚持过去的假设，有时在面临着对过去假设否定的压倒性的证据面前亦是如此(Jng 2006）。同样，股票投资获得收益的投资商会根据经验舒缓地释放神经素多巴胺，帮助他们对市场的行为做出决定(L和Repin 2002）。

我们暂时假定你既不是科学家也不是股票投资商，而只是一个渺小的调查员，对以下两个投资方案选项做出选择：

·选项A：100%收益7400美元

·选项B：75%可能收益10000美元，25%没有任何收益

这里没有什么技巧：你可以看到每种结果的期望效益。你会选择哪一种呢？

现在，给出以下两种选项，你又会选择哪一种呢？

·选项C：100%损失7400美元

·选项D：75%可能亏损10000美元，25%没有亏损

绝大多数人选择A和D。这里出现了不对称的情况：我们往往在下跌时更愿意冒险，而在上涨时则更保守。也就是说，稳赚比赚得更多但可能不赚更吸引人，但是当需要支付一定资金时，我们宁愿选择可能亏损更多可能不亏的情况。此外，这种投资组合并没有得到最大回报：如果我们只是投机性地考虑收益和损失，我们应该选择B+C的组合方式，因为其总期望收益是1000美元，而A+D组合方式的总期望收益是-1000美元（也就是说，比起A+D组合方式，B+C组合方式总期望收益高出2000美元）。

Daniel Kahneman和Amos Tversky在20世纪70年代首先做了该实验，它揭示了人们通常都不是以概率的方式思考问题：相反，我们想象和每一种单一收益相关的感情收益。

事实上，在很多重要的方面，我们并不是以自己所假定的方式来处理数据。

何时数据无法驱动

前面的内容指出了人们在分析数据时存在的一些认知偏见。本文的剩余部分讨论数据所不能做的，也就是说，其衡量和解释本身可以转换数据的各种方式。本文不是一篇关于“谎言、天杀的谎言和统计”的论文：我们知道数据可以被故意混淆；本文的重点是它如何会被无意混淆。特别是在如下情况：

·我们使用数据的方式不够准确

·我们采用已知的偏见方式处理数据

虽然以下例子重点研究医学和财务数据，但是它们或多或少可以扩展。“数据”用于表示任意从经验、观察和实验中积累的原始的事实。

数据并非越多越好

统计是一门表示和近似的科学。我们捕获或者观察一个系统越多，就越能真实地表示它。一篇入门性的文章往往会强调：随着你增加样本大小，置信区间就减少，而没有丧失任何置信度。换句话说，更多的数据可以帮助你控制误差边际（见图13-4）。

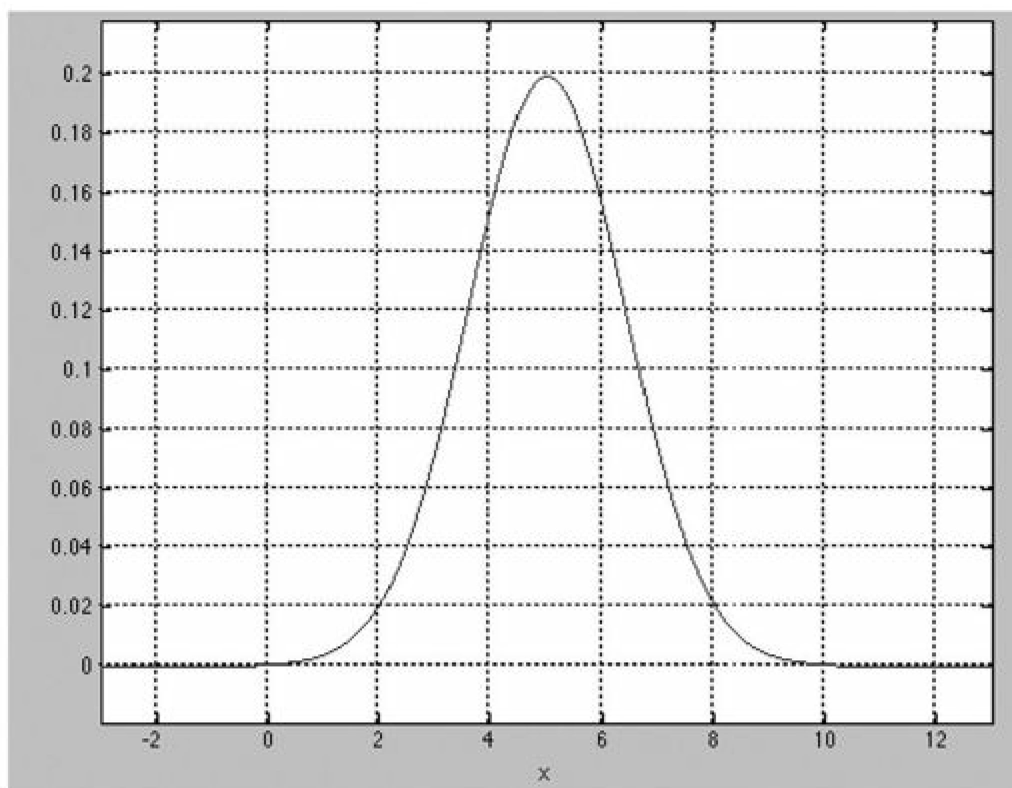


图　13-4：正态分布（见彩图49）

教科书上一个不错的真相。在游离世界外，需要审查一些假设。首先，数据是如何分布的？它是否是正态的？例如，很多财务数据的分布偏离常态。医学数据（例如，特征表达）通常更多的是呈高斯分布，但是演变并不总是符合中心极限定理。

如果数据不是正态分布，更多的数据将不会减少你期望的误差边际。Karl Popper描述了我们如何使用数据来回答问题上的偏态：

虽然无论有多少个与假设一致的结果都无法证明该假设是正确的，但是只要存在一个反面结果，就可以推翻该假设。更多的数据只是增加了必然性边际，而一个实例就可以推翻一个世纪的信仰。

其次，“误报率”(flse positive)和“漏报率”(flse negative)的代价相同

吗？即使你的数据是（或者看起来）是正态分布的，你对不同结果的兴趣可能也是不对称的。

例如，无法检测到威胁生命的疾病的误差所付出的成本可能比错误的诊断更高。在这种情况下，（通过减少“漏报率”）提高诊断正确性的数据比通过大量数据来减少“误报率”更有用。

更多的数据并不容易

数据并不一定需要大规模。信息时代一个陈腐的“箴言”是：处理10TB的数据和处理10比特的数据一样简单，而制作100亿个向导小工具要比制作10个更昂贵。

在某些情况下，清洁和处理数据的代价不低。当需要人工肉眼校验，比如阅读X射线或者问卷上的手写编码数据时，尤其如此。依据“红桃皇后”(Rd Queen)效应[1]，性能更好的计算机及其愈来愈强的收集数据的能力驱动了（而且被驱动）开发新的工具和采用新的方式来解析和使用数据。

也存在包含更多信息的认知成本。我们是选择超市的拥挤还是“401(k)条款”（401(k)plans)[2]，研究表明随着选项的增加，决策需要花费的时间越来越多，我们变得更可能不做任何选择而直接放弃，而且对于自己做出的选择的满意度也降低了(Ienger和Lepper 2000）。

最后，一个微妙的成本是：更多的数据会使我们变得盲目而看不到其他的可能性，尤其当我们需要收集并整理数据时。很难想象，看到更多的数据意味着更好地支持一个假设——确认偏差和抽样问题的必然结

果之前已经讨论过了。

数据不会自我解释

人们做出了种种解释。你可能已经听说，相关性(crrelation)和因果关系(cusality)是“聚头冤家”(srange bedfellows)。给定两个统计上密切相关的变量，因果关系既可以变大也可以变小。统计学家喜欢滥用相关性（更不用提很多博客了），比如对现代世界传统价值观的衰落有所不满的老太太。

记者是这种统计“不满”的首要主导因素。例如，《华尔街日报》最近的一篇文章指出(sellenbarger 2008），由于婚前同居离婚的概率更高，未婚夫妇为了提高婚后呆在一起的概率，应该避免婚前同居。该项研究没有任何迹象表示其中存在因果关系，而这个记者基于“数据”，给情侣们提供了她自己的建议。

用因果关系替代相关性不需要如此明显。当开展一项科学研究项目时，存在以下假设：如果发现事物间存在相关性，就意味着它们存在因果关系，否则其关系就是未知的。否则，为什么要去回答研究问题呢：大规模地搜索没有因果关系的相关性属于偶然性计算，不是科学。即使是所谓的大数据，科学依然是一个很强的由假设驱动的过程。

经验研究的局限不是有道理的服输，只是小心谨慎地推进研究发现，不受因果关系影响。基于数据创造故事是人之常情：研究是不断地修改故事，使它变得合理。

仅仅是答案的数据是不好的

描述性统计信息会隐藏细节信息。例如，图13-5的图表显示了看起来显著不同的四种分布，但其均值和方差却是相同的。描述性统计学的两大支柱——均值和方差——提供的关于分布的信息很少(Ascombe 1973）。

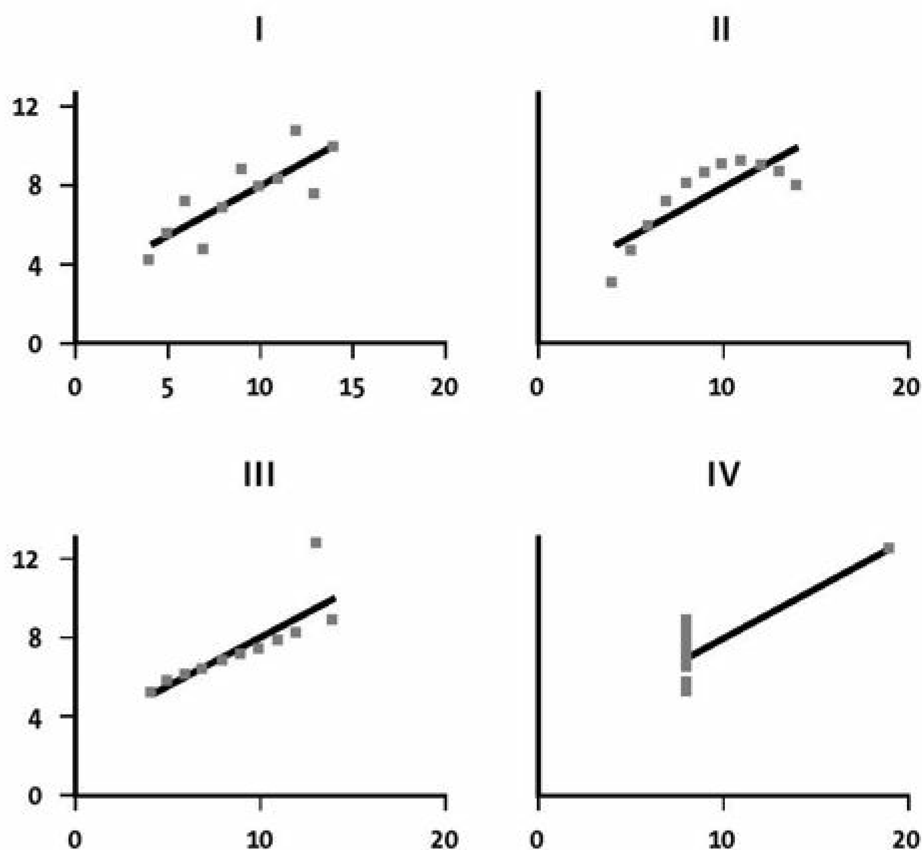


图 13-5：Anscombe的“四重奏”(qartet)：每组数据集都有相同的均值和方差

当使用数据来制定决策时，我们倾向于认为数据分布只是为了最后的一个答案。我们可能需要基于二选一来制定决策——美国应该宣战吗？FDA（美国食品药品管理局）应该批准这种药物吗？预计谁会赢得竞选？（或者是做一个总结性陈述）美国人有多富有？在今后5年，地

球的气候将会如何？——这些都是基于不确定性数据。即使报告了方差，重要的还是决策。

人们在思考时，考虑的是结果，而不是分布。以个人财务决策为例：我应该在股票、债券和现金上投资多少？即使过去的财务业绩可能预测将来的回报（即使财务顾问法律上需要承认这一点，事实上也并非如此）——也就是说，即使我们知道了分布的状态——我们还存在很多可选择的风险和回报组合，以及在这些分布范围内的很多可能的结果。给定一个风险水平，一个人退休时可能很富有也可能很贫穷，同时想象这几种情况很难（人们倾向于假设一个平均情况，或者有时是最佳的情况，即所谓的“规划性谬误”(panning fallacy)）。

一个决策学科学家团队创建了一个有趣的工具来帮助投资者理解结果分布的固有的概率范围（见图13-6）。参与者可以调整100个“单位概率”来形成一条分布曲线。举个例子，他们可以把所有单位都放在75%的薪金的位置，或者把这些单位平均分发到很多不同的比例水平。然后，点击“开始”(g)，观察这些单位一个个逐渐随机消失。最后留下的一项是“结果”(otcome)(Gldstein等2008）。因此，风险水平不是一条模糊的分布曲线，而是一组（这里是100个）具有相同发生概率的曲线。

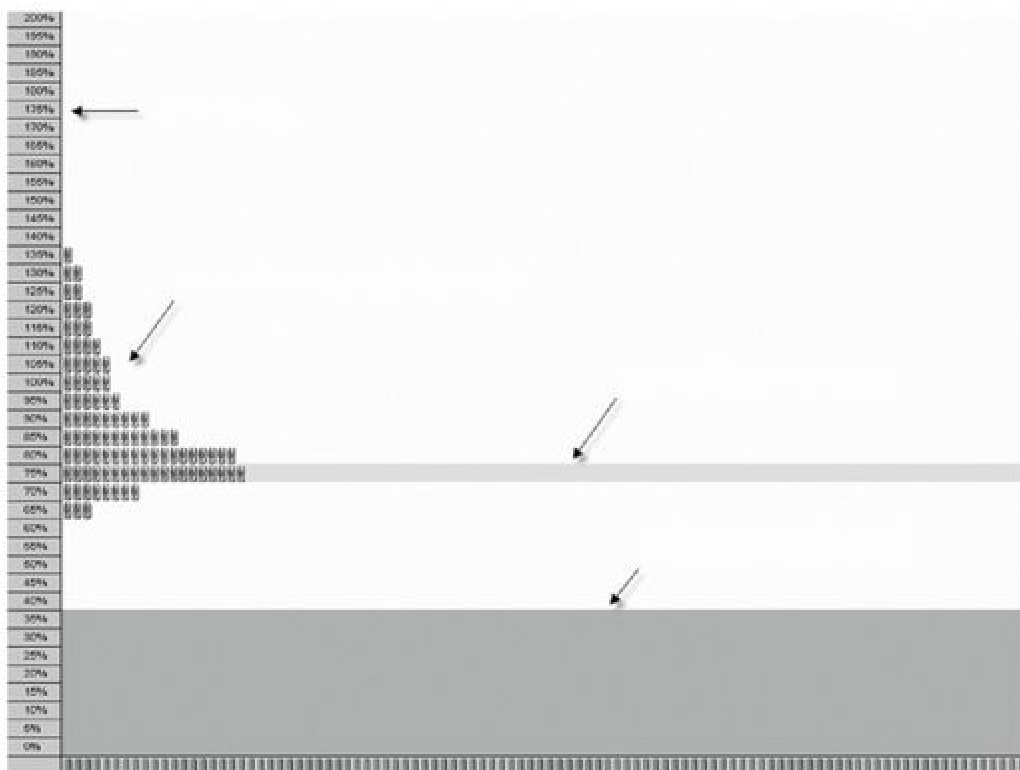


图　13-6：Goldstein等开发的工具帮助人们理解作为一组结果的分布（见彩图50）

生物学家Stephen Jay Gould通过和结果等同的描述统计进一步阐明了这个问题。“中间产物不是想要的信息”(Te Median is not the Message)是Gould对癌症诊断和“只能活8个月”的警告的反应。关于癌症的文章基于“预先定义的一组环境”揭示了向右偏分布——也就是在过去治疗情况的假设上，寿命很长的幸存者的一个很长的故事。声称“8个月”的生存期具有很大的局限性，正如Gould优雅地给出兽性的统计特征：

进化生物学家知道自然变异本身是唯一不可改变的本质。变异是残酷的现实，而不是对集中趋势的不完美的衡量尺度。均值和中位数是抽象化的。

数据不能预测

建立模型（来预测明天的天气，2012年超级杯的结果或者世界500强的命运）是一门很吸引人的艺术。事实上，科学最重要的探险——解释我们周围的世界的一个重要扩展，正试着去了解世界的未来。

在某些领域，即受控的物质世界宇宙，有可能预测几乎确定的结果。未来的结果可以高保真地从过去事件的发展轨迹中得出：水加热时会变成气体；物体在真空中会以每秒9.8米的加速度下降；如果动物的心脏停止跳动，它就死了。从认识论上看，Popper的“可否认性”(flsifiability)理论没有实际意义，但是如果之前三个假设是真实的，那么它可以引导出健全的社会生活。

在确定性较低的领域，比如人类或物理行为，模型是帮助解释模式的重要工具。但是，当我们过于渴望让数据说话时，可能会导致生成过拟合模型。

以发现多普勒径向速度为例（该过程我仅有表面简单的理解：基本上，明亮的星星使得查看行星变得很难，因此天文学家识别了多普勒变化的各种组合，它们只有在行星绕过恒星时才会出现）。很难检测模型的灵敏度，而且只有15个观察，其数据可能非常适合图13-7的正弦曲线(G等2004）！

当模型过拟合时，它就失去了预测能力。此外，如果我们喜欢接受任何最适合已有数据的模型，而不关心其复杂性或灵敏性，那么我们就会犯一些错误。首先，忘记数据的因果关系，对它有损害，过度调试的

模型无法说明任何东西。

其次，我们忘记数据（或者数据集合）可能是有限的，而世界本身可以改变。以试图预测200年前的世界气候问题为例。存在一些关键证据，长期以来还保持很高的分辨率，即化石记录和冰核中的全球气温数据。气候学家还可以推断出当地气温和来自日记和树的年轮的沉淀，但是其精度水平差别悬殊：18世纪的暴风玻璃和20世纪带GPS的气象气球不同。而且，谁能够知道相同组合的交互在21世纪驱动的气候事件和20世纪会相同呢？

同样，1914年的福特汽车公司和1975年不一样，和今天的福特也不一样，但是很多金融模型假定市场的最后一个周期的动态性也将会解释其未来的业绩（而且，模型对要考虑的相对时间周期做出了非常不同的假设）。因此，风险分析模型可能在很多时候只是偏离（可接受程度）一点点，但是当发生“未预期”的情况时，它可能就会被完全打破（例如房地产市场崩溃）。

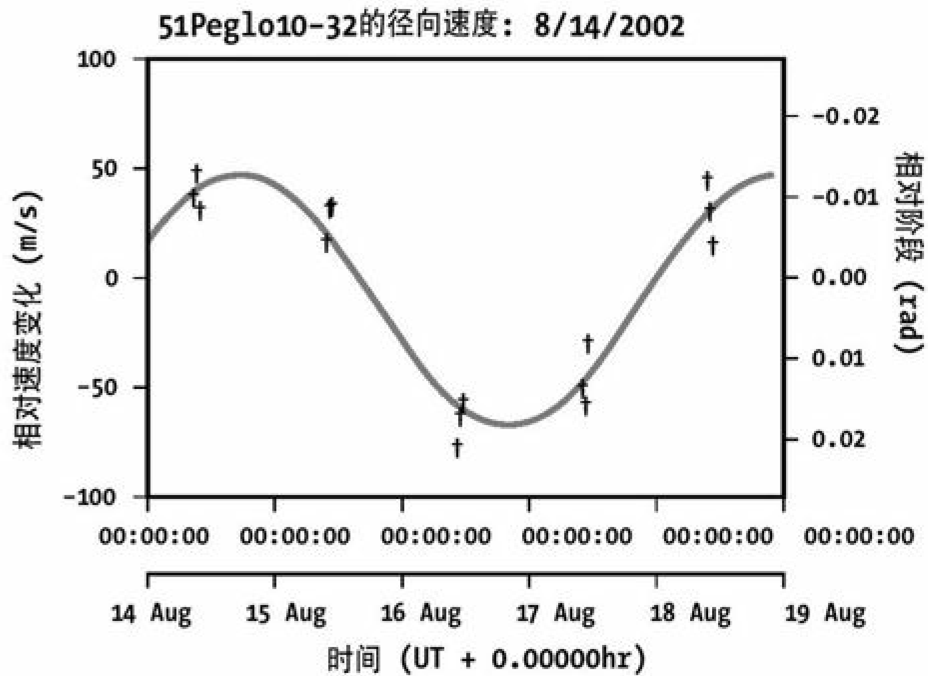


图　13-7：外星识别模型

好的科学家了解坏模型的危险，但是被一个看起来特别好的模型所“诱惑”也不难。举个例子，2005年报告的过拟合模型的一个单位的经验（它当时是正确的，但是事后看来，它还差得很远）：

当一种新的模型方法大大提高了能量时，人们对它产生一定的怀疑是合适的。这种新方法通常是拟合数据收集的结果，而不是实际的底层行为关系。此外，这些问题在事后看来很显然，但是在我们当时有的数据清理过程中，并没有出现这些问题……最好的情况是，过度拟合会给新模型带来不必要的复杂性或者用户会诋毁该模型。最坏的情况是，它导致投资组合的风险评估的系统错误。

看看吧！

存在很多令人信服的理由来建立预测模型，包括探索场景和阐明假

设；一个优秀的例子，参考Joshua Epstein 2008年发表的文章“为什么建模？”

概率不是直观的

这是构建统计时的另一个招人喜欢的“鞭笞”(fogging horse)策略，而且是出于善意的。统计学家不知疲倦地设计可爱的游戏，证明一个看似常识性的答案不正确的概率，而且条件概率和联合概率不是直观的。当数学家们和医生们被这些游戏愚弄时，这些统计学家便会格外高兴。

给定一个美国城市，100万居民当中大约有1000名（或0.1%）居民携带艾滋病毒阳性。一种新的艾滋病毒测试，包含1%的误诊率：每一百次中会有一次，它会将一个艾滋病毒阴性的人误诊为艾滋病毒携带者；反之亦然。

假设有人参加了测试，并被诊断携带艾滋病毒阳性。那么，他患有艾滋病毒的概率是多大呢？

很多人会回答说他患有艾滋病毒的概率是99%，因为测试的误诊率是1%。实际上，因为患有该疾病的人口比例非常小，任何人患有该疾病的概率，即使被诊断患有，也只有9.9%。（99.9万的艾滋病毒阴性的居民，有9990个会被诊断为携带艾滋病毒，而其中只有990个真正有艾滋病毒。对于结果是携带艾滋病毒的诊断，实际上真正有艾滋病毒的概率是990/9990或9.9%。）

医生给出的诊断错误率很高，至少是值得怀疑的。

在很多情况下，先验不会消失。当使用数据来回答问题时，我们不

知道应该排除哪些证据，以及如何权衡哪些证据更重要。Daniel Kahneman——一个不知疲倦的概念命名者——把这种现象命名为“基准概率谬误”(bse rate fallacy)。

概率事件不是直观的

不但概率论很难把握，每个概率事件也很飘忽不定。当缺乏因果关系来把一个事件绑定到一组结果集上时，人们依赖过去的观察来估计概率。而且观察通常是以偏见方式收集的（尤其当通过经验囊括，而通过实验也很平常），并且很难记录、调试、加权、保存和查询。

真实世界并不创建随机变量

最开始，地球是没有形状的，而且没有东西。然后Fisher说：“让我们有z分数和变异数。”于是存在了z分数。Fisher看到回归很好，于是他把有意义的和无意义的区分开来。统计学的发明看起来是如此重大的创新，但是它们很难记住，因为它们不是自然规律。人可以想象另一个宇宙，那里设置了统计意义的标准阈值（被任意设置，正如在我们宇宙里）为p=0.01或p=0.06，而不是当前的p=0.05。想想那些已经被批准或拒绝的毒品、环境变量和健康影响之间的错误关联，你在汽车保险上花费的大量金钱!

在没有Fisherian的世界，不存在类似于独立随机变量的东西。实际上，很多东西关联度都很高。目前，实验控制相互依赖性是可能的，但是独立性就很难保证。正如我们最近所知，假设离散事件（如购房贷款人按揭拖欠）相互独立可能是错误的，但却有很多理论基于这种假设

（如可交易的金融产品融入了贷款款项）不一定是错误的。

预测市场和群体决策制定过程能够非常出色地工作——在某些情况下，优于一组专家的估计。然而，当信息瀑布和相互依赖关系进入系统时，它们已经证明被打破了(Bkhchandani等1998）。

数据不是独立的

在现实世界的决策制定中，数据有很多种形式。信息很少会被清理并打包成一个格式良好的电子表格或“矩阵文件”(mtrix file)；取而代之，我们通常需要基于主观以及量化信息来制定决策。

举个例子，是否制定在线把钱借给别人的决策（为了盈利，作为建立贷款市场的一部分）。我和同事使用一组来自于P2P(peer-to-peer)平台Prosper.com的35万贷款数据集，对资金借贷和偿还进行了分析，其结果表明不论有多少个模型（混合模型、神经网络、决策树和回归）能够预测谁能够获得贷款以及谁能够准时偿还的准确率只有75%。大量的数据——包括该网络的每个人有100多种个人财务健康指标——都可以根据现有算法来计算，但是哪个申请者可以获得资格以及哪个无法得到资助还是难以预测。通过量化主观特征，模型可以得到部分完善。当一个人决定是否把钱借给网络的某个成员时，出借人（和银行不同）考虑了很多“软”因素：借款人的目的声明、公司形象、拼写、语法以及其他个人资料信息。为了把其中一些特征纳入我们的模型，我们请人工（来自Amazon的Mechanical Turk)对Prosper.com网站的成员图像进行编码，首先是内容——该图像是否描述一个人、家庭、车等——然后是“诚信”评

分：即该问题的答案，“你会借钱给这个人吗？”

但是模型仍然存在不足：社会因素对贷款的动态性有着意想不到的作用。和我们的假设相反，贷款给别人的决定不是独立的。而证据表明，竞标存在一些“羊群效应”：出借人学其他出借人，而且随着一个贷款上的竞标增加，单位时间的竞标也加速。

即使考虑这些情况以及其他的社会因素，很多出借人还是制定了次优的贷款决定。Proper网站在理论上是一个接近完美的市场：几乎所有的人都可以访问该站点的API，并重复我们的分析。但是出借人对于非常冒险的投资，还是只得到非常低的回报：给定统计的期望回报，很差的投资赌注的数量多得惊人。即使有可靠的信息（而且主观数据代理人谦逊温和），决策通常不是根据数据直接指定的，反过来，数据只能够从一定程度上解释人们的决策。

数据受到观察者的影响

最后，即使在可靠的因果关系是可能的情况下，如果数据是由Fisher和Bayes（如果该学生如此虔诚）的一个明智的学生老老实实地收集和建模，并由该学生负责改变并验证其模型（而且对模型结果仍然持怀疑态度），那么很多认知偏见会模糊人们的思考。在真实世界中，我们最好能够以概率性假设方式来操作。

正如统计学家常常在他们的博客上表示不满，行为经济学家在各自的编年史上都臭名昭著。如之前提到的叙述性谬误、确定偏见、选择悖论、冒险、基础概率谬误以及夸张的贴现。心理学家已经索引了很多其

他的，包括从“锚定”（过度依赖于最近单点数据来制定决策）到“Lake Wobegon效应”（超出一半的人认为他们高于平均水平）。

随着这些影响被更好地记录，我们可以开发工具并利用直觉来帮助采用数据的票面价值（我的一部分工作重点为金融决策开发工具）。从某种意义上来说，解决办法很简单：如果你不理解数据的局限性，那么数据并没有太多价值。

[1]Lewis Carroll的《爱丽丝梦游仙境》中的“红桃皇后”说“为了呆在原地，你需要尽力奔跑”。该思想被用于描述一个系统，由于外部压力的竞争，必须不断地和竞争者共同演化发展。

[2]401(k)条款指的是20世纪80年代初制定的美国《国内税收法》的第401条第k项条款，是由雇员和雇主共同缴费建立起来的养老保险制度。

结束语

不用多说，我们在一个丰富的数据时代生活。很多东西便宜松垮。进化过程已经使我们能够注意到环境的显著变化——众所周知的老虎和海啸——识别老虎的脸，描述海啸以帮助记忆。但是我们缺乏基础设施来收集和排序大量的异构数据集。我们有哪些基础设施必须慎重利用：我们可以通过理解概率和概率的极限来开始更好地利用数据，而且对帮助阐释世界的认知偏见持审慎态度。

观察者眼里的数据真的可以很美！

参考文献

[1]Anscombe,F. J."Graphs in Statistical Analysis."The American Statistician,vol.27，no.1(Fbruary 1973），pp.17-21.

[2]Bikhchandani,S. et al.（1998）."Learning from the Behavior of Others:Conformity,Fads,and Informational Cascades，"Journal of Economic Perspectives,vol.12，issue 3，pp.151-170.

[3]Dwyer,D. W.2005."Examples of overfitting encountered when building private firm default prediction models"(ww.moodyskmv.com/research/files/wp/Overfitting_Private_Firm_Models.pd

[4]Epstein,Joshua M.（2008）."Why Model?". Journal of Artificial Societies and Social Simulation 11（4）12.

[5]Ge,J. et al.，2004，"All Sky Extrasolar Planet Searches with Multi-Object Dispersed Fixeddelay Interferometer in Optical and near-IR."Proc.SPIE，5492，711.

[6]Goldstein,Daniel G. et al.（2008）."Choosing Outcomes Versus Choosing Products:Consumer-Focused Retirement Investment Advice."Journal of Consumer Research，35(Otober)，440-456.

[7]Gould,S. J."The Median is not the Message."Discover Magazine，1985.

[8]Iyengar,S. S.and M.R.Lepper."When choice is demotivating:can one

desire too much of a good thing?"Journal of Personality and Social Psychology，2000，vol.79，no.6，995-1006.

[9]Jeng,M."A selected history of expectation bias in physics."American Journal of Physics，2006.

[10]Kahneman,Daniel and Amos Tversky（1979）."Prospect Theory:An Analysis of Decision under Risk，"Econometrica,XLVII（1979），263-291.

[11]Krumme,C."Telling tales:the effects of narrative creation on decision-making with data，"working paper.

[12]LO'A. W.and D.V.Repin."The Psychophysiology of Real-Time Financial Risk Processing"Journal of Cognitive Neuroscience,April 1，2002，vol.14，no.3，323-339.

[13]Lord,C. G.et al.（1979）."Biased assimilation and attitude polarization:The effects of prior theories on

[14]subsequently considered evidence."Journal of Personality and Social Psychology，37，2098-2109. Shellenbarger,Sue.Wall Street Journal"Work and Family"column,March 22，2008.

第14章　自然语言语料库数据

Peter Norvig

本书大部分阐述的是“波德莱尔(Budelaire)意义”上的数据之美：“一切美好和崇高的都是理性和计算的结果。”本章的数据是“梭罗(Toreau)意义”上的数据之美：“人们总是被最平凡的演讲之美打动。”本章要阐述的数据是“最平凡的演讲”：取自公开的Web页面中的总长度达1MB英语单词的数据。这些数据涵盖了Web所有的“陈词滥调”，既有拼写和语法错误、哈哈大笑(Lugh Out Loud,LOL)的猫、踢踢滚滚(Rckrolling)，也包含马克·吐温、狄更斯、奥斯汀和几百万其他作家的作品集。

G公司的Thorsten Brants和Alex Franz于2006年发布了1MB的单词数据，你可以在语言数据联盟(Lnguistic Data Consortium)上可以获取(htp：//tinyurl.com/ngrams)。该数据集通过计算每个单词的出现次数，依照每二词(to-word)、三词、四词和五词序列对原始文本求和。举个例子，“the”这个单词出现了230亿次（占了1MB单词的2.2%），该词是最常用的单词。单词“rebating”出现了12750次（占百万分之一），还有“fnuny”（显然是“funny”的误拼）。在三词序列中，“Find all posts”出现了1300万次（占0.001%），“each of the”出现频率与之相似，但都低于出现了1亿次的“All Rights Reserved”（占0.01%）。以下是三词序列的一个摘录：

```
outraged many African 63
```

```
outraged many Americans 203
outraged many Christians 56
outraged many Iraqis 58
outraged many Muslims 74
outraged many Pakistanis 124
outraged many Republicans 50
outraged many Turks 390
outraged many by 86
outraged many in 685
outraged many liberal 67
outraged many local 44
outraged many members 61
outraged many of 489
outraged many people 444
outraged many scientists 90
```

从摘录可见，土耳其人是最易产生不满的组织（来自网络，根据当时收集的数据），共和党和自由党人有时会产生不满，而民主党和保守党则很少有不满。

为什么我说数据很美并不乏味？单个计数都是乏味的。但是这些计数的聚集——几亿个计数则是美丽的，因为它具有很多含义。计数的聚集不再仅仅是英语，而是关于说英语的人们的世界。数据是美丽的，因为它表示了很多值得表达的东西。

对于这些数据，在观察可以做什么之前，我们需要弄清如何探讨(tlk the talk)——学习一些术语。文本的集合称为语料库。我们把语料库看做token序列——单词和标点符号。每个不同的token称为类型(tpe)，因此文本“Run,Lola Run”包含四个token（逗号也作为一个token)，但只有三种类型。类型的集合称为词汇。G公司语料库包含1MB的token，1300万种类型。英语在词典里出现的单词大约只有100万，但是语料库包含这样的类型，

如“www.njstatelib.org”、“+170.002”、“1.5GHz/512MB/60GB”和“Abraham
但是绝大多数的类型是很少见的，最通用的10个类型占几乎1/3的token，最通用的1000个占超过2/3，而最通用的10万个占98%。

1-token序列是一元，2-token序列是二元，n-token序列是N元。P代表概率，如P(the)=.022，表示token“the”的概率是0.022或者2.2%，如果用W表示token序列，那么W3表示第三个token，而W1：3表示从第一个到第三个token。P(Wi=the|Wi-1=of)是token“the”的条件概率，表示在先验token“of”下的概率。

G公司语料库的一些细节：出现次数少于200次的单词作为未知类型，以符号＜UNK＞表示。丢弃出现次数少于40次的N元。该策略减少了打字（错误）的影响，使得数据集只有24G（压缩后）。最后，语料库的每个句子都以特殊的字符＜S＞开始，＜/S＞结束。

我们现在来看看可以利用该数据完成的一些任务。

分词

考虑中文文本“浮法像蝴蝶”，它的英语翻译是“float like a butterfly”。它包含5个字符，但是字符间没有空格，因此中文阅读者需要执行分词：决定单词的边界位置。英语阅读者通常不需要执行分词，因为单词间存在空格。但是，一些文本，如URL，通常不包含空格，但是有时输入错误，会留个空格；搜索引擎或文本处理程序如何纠正这样的错误呢？

比如英语文本“choosespain.com”，这是个网站，希望你选择西班牙作为旅行终点站。但是如果你把名字切分错误，你会得到这样的分词“chooses pain”，这个名字就不那么吸引人了。人工阅读可以通过几年的经验来做出正确的决策；但是把这种经验编码为计算机算法将是一项难以完成的任务。但是我们可以采取一种捷径，其效率之好让人惊讶：在二元表里查询每个词汇。我们发现“choose Spain”出现了3210次，而“chooses pain”在表里一次都没有出现（这意味着它在1MB的语料库中出现次数少于40次）。因此，“choose Spain”出现至少80次，可以作为正确的分割。

假如我们需要解释短语“insufficientnumbers”的含义，如果我们把单词的大小写加起来，其计数是：

```
insufficient numbers 20751
in sufficient numbers 32378
```

“In sufficient numbers”的出现频率高于“insufficient numbers”，但是这并不具备很强的说服力。这种情况令人沮丧：我们可以猜测，但是无法确定。对于类似这种不确定问题，没有方法能够确定地给出正确的值，我们还没有能够确保答案正确的完备模型，实际上人类专家也没有一个完备的模型，可以不同意该答案。但是，解决不确定问题有个确定的解决方案。

1.定义概率模型。我们无法定义所有的因子（语义的、语法的、词汇的和社会的）使得选择“choose Spain”对于一个域名是一个更佳的候选。但是我们可以定义一个简化的模型，从而能得到近似概率。对于“choose Spain”这种简短的候选，我们可以在语料库数据中查找n元，并使用该概率。对于更长的候选，我们需要切分为多个部分，组合这些部分得到结果值。核心是我们定义语言模型——在该语言中所有字符串的概率分布——从语料库数据中学习该模型参数，然后使用该模型来定义每个候选的概率。

2.枚举候选项。我们无法确定“insufficient numbers”还是“in sufficient numbers”更可能是所期望的短语，但是我们可以确定它们都是候选分割，如“in suffi cient numb ers”也是候选项，但“hello world”不是有效的候选项。在该步骤中，我们不做判断，只是枚举可能的候选项——如果可以，列出所有的候选项，或者列出仔细选择的一个样本。

3.选择最可能的候选项。对每个候选项应用语言模型来获得它的概率，选择概率最高的是选项。

如果你更习惯于数学等式表达，如下所示：

best=argmaxc∈candidates P(c)

或者，如果你更习惯于计算机代码表达，则如下：

```
best=max(c in candidates,key=P）
```

我们把这种方法用于分割中。定义一个函数、分段，把不包含空格的字符串作为输入，返回最佳分段的单词列表。

```
＞＞＞segment（'choosespain'）
['choose'，'spain']
```

我们从步骤1——概率语言模型开始。单词序列的概率是每个单词概率的乘积，假设单词的上下文是：所有上述单词。等式表达如下：

P(W1：n）=Πk=1：nP(Wk|W1：k-1）

我们没有数据来精确地计算该等式。因此我们可以使用一个更小的上下文来近似计算。由于数据序列大于五元，我们就采用五元文法模型，因此N元文法的概率即给定前缀四个单词（不是前缀所有单词）的每个单词的乘积。

五元文法模型存在三个难题。第一，五元数据大约30G，因此无法全部装载到内存中。第二，很多五元计数将为0，我们需要一些回退(bcking off)策略，使用更短的序列来估计五元文法的概率。第三，候选项的搜索空间将会很大，因为依赖性扩展到四个单词。这三个难题花些时间都是可控的。但是，我们首先来考虑解决这三个问题的一个更简单

的语言模型：一元模型，其序列概率即每个单词自身的概率乘积。在一元模型中，每个单词的概率和其他单词无关。

P(W1：n）=Πk=1：nP(Wk)

对'wheninrome'进行分割，考虑这些候选项，如when in rome，然后计算P(when)×P(in)×P(rome)。如果该候选项的乘积高于任何其他候选项，那么它就是最佳答案。

n个字符的字符串包含2n-1个不同的分割方式（字符间有n-1个位置，每个位置都可以作为分割边界）。因此，字符串'wheninthecourseofhumaneventsitbecomesnecessary'包含35T的分割方式。但是我确定你可以在几秒钟内找到合适的分割方式；显然，你不可能枚举所有的候选类型。你可能浏览了“w”、“wh”和“whe”，并作为不可能的单词放弃它们，而接受“when”作为可能的分割方式。然后继续剩余部分，并找到它们的最佳分段方式。一旦我们简化假设每个单词和其他的单词是不一样的，那意味着我们不需要考虑单词的所有组合。

以上简单地给我们描述了segment函数：考虑把文本划分为一个起始单词和剩余文本的所有方式（可以任意限制单词最大长度，如L=20个字母）。对于每种可能的划分，找到对剩余文本分词的最佳方式。在所有的候选项中，乘积P(first)×P(remaining)值最高的即为最佳方案。

这里我们用一个表格来说明这个问题。表格中包含了第一个单词的所有候选、该单词的概率、剩余单词的最佳分词概率以及所有的概率（即第一个单词和剩余单词的概率的乘积）。我们发现，以“when”开始

的分词概率是第二个最佳候选的5万倍。

first	P(*first*)	P(*remaining*)	P(*first*) × P(*remaining*)
w	$2 \cdot 10^{-4}$	$2 \cdot 10^{-33}$	$6 \cdot 10^{-37}$
wh	$5 \cdot 10^{-6}$	$6 \cdot 10^{-33}$	$3 \cdot 10^{-38}$
whe	$3 \cdot 10^{-7}$	$3 \cdot 10^{-32}$	$7 \cdot 10^{-39}$
when	$6 \cdot 10^{-4}$	$7 \cdot 10^{-29}$	$4 \cdot 10^{-32}$
wheni	$1 \cdot 10^{-16}$	$3 \cdot 10^{-30}$	$3 \cdot 10^{-46}$
whenin	$1 \cdot 10^{-17}$	$8 \cdot 10^{-27}$	$8 \cdot 10^{-44}$

我们可以通过几行Python代码来实现分词：

```
@memo
def segment(text):
"Return a list of words that is the best segmentation of text."
if not text:return[]
candidates=（[first]+segment(rem)for first,rem in splits(text)）
return max(candidates,key=Pwords)
def splits(text,L=20）:
"Return a list of all possible(first,rem)pairs,len(first)<=L."
return[(txt[: i+1]，text[i+1: ]）
for i in range(min(len(text)，L）） ]
def Pwords(words):
"The Naive Bayes probability of a sequence of words."
return product(Pw(w)for win words)
```

这是整个程序——包含三个较小的省略部分：product是把数字列表进行乘积的工具函数；memo是一个装饰器(dcorator)，对函数product的结果进行缓存，因而这些结果就不需要重新计算；Pw通过询问单元计数数据来估计一个单词的概率。

没有装饰器memo，对一个包含了n个字符的文本段的调用会导致对该段的2n次递归调用；有了memo，它就只执行n次调用——memo使得调用变成一个高效的动态编程算法。对于n次调用，每个都需要考虑O(L)次的分片，对每个分片乘以O(n)的概率来估算每个分片的代价，因

此整个算法代价是$O(n^2L)$。

对于P w，我们从一个数据文件读取单元计数值。如果一个单词在语料库中出现，它的估计概率是Count(word)/N，其中N是语料库的大小。实际上，我没有采用1300万类型的单元数据文件，而是创建了vocab_common，它有几方面特性：（1）大小写不敏感，因此“the”、“The”和“THE”的计数值都加在一起，作为一个词条的“the”；（2）词条是由字母表示的，而不是数字或者标点符号（因此“+170.002”不能作为词条，同样“can't”也不能；（3）100万单词中最通用的1/3单词。

Pw唯一较难处理的部分是当一个单词不在语料库中时如何处理更为合适。即使是在包含了10000亿个单词的语料库中，这种情况有时也会发生，因此把概率当做0返回是错误的。但是它应该为多少呢？语料库中的token个数N，几乎有10000亿，vocab_common中最不经常出现单词有12711个。因此之前没有看见的单词，其概率应该在0～12711/N。不是所有未见到的单词的概率都是一样的：一个有20个字母的随机序列是单词的概率要小于一个有6个字母的随机序列。我们将为概率分布定义一个类Pdist，它加载一个内容是(ky,count)对的数据文件。默认情况下，一个未知单词的概率是1/N，但是Pdist的每个实例可以提供一个自定义函数对默认值进行重载(oerride)。为了避免对于很长的单词概率太高，因此我们确定（相当随意）了一个起始概率10/N，对于候选单词中的每个字母，以因子10来递减。我们定义Pw为一个如下的Pdist：

```
class Pdist(dict):
"A probability distribution estimated from counts in datafile."
def__init__(slf,data,N=None,missingfn=None):
for key,count in data:
self[key]=self.get(key, 0) +int(count)
self.N=float(N or sum(self.itervalues () ) )
self.missingfn=missingfn or(lambda k,N: 1./N)
def__call__(slf,key):
if key in self:return self[key]/self.N
else:return self.missingfn(key,self.N)
def datafile(name,sep='\t') :
"Read key,value pairs from file."
for line in file(name):
yield line.split(sep)
def avoid_long_words(word,N) :
"Estimate the probability of an unknown word."
return 10./ (N*10**len(word))
N=1024908267229##Number of tokens in corpus
Pw=Pdist(datafile ('vocab_common') , N,avoid_long_words))
```

注意，Pw[w]是单词w的原始计数，而Pw(w)是它的概率。本章描述的所有程序都能够通过http：//norvig.com/ngrams获得。

因此，该模型分割的效果如何呢？以下是一些实例：

```
>>>segment ('choosespain')
['choose', 'spain']
>>>segment ('thisisatest')
['this', 'is', 'a', 'test']
>>>segment ('wheninthecourseofhumaneventsitbecomesnecessary')
['when', 'in', 'the', 'course', 'of', 'human', 'events', 'it', 'become
>>>segment ('whorepresents')
['who', 'represents']
>>>segment ('expertsexchange')
['experts', 'exchange']
>>>segment ('speedofart')
['speed', 'of', 'art']
>>>segment ('nowisthetimeforallgood')
['now', 'is', 'the', 'time', 'for', 'all', 'good']
>>>segment ('itisatruthuniversallyacknowledged')
['it', 'is', 'a', 'truth', 'universally', 'acknowledged']
>>>
segment ('itwasabrightcolddayinaprilandtheclockswerestrikingthirteen')
['it', 'was', 'a', 'bright', 'cold', 'day', 'in', 'april', 'and', 'the
```

```
'were', 'striking', 'thirteen']
>>>
segment('itwasthebestoftimesitwastheworstoftimesitwastheageofwisdomit
offoolishness')
['it', 'was', 'the', 'best', 'of', 'times', 'it', 'was', 'the', 'worst
'it', 'was', 'the', 'age', 'of', 'wisdom', 'it', 'was', 'the', 'age',
'foolishness']
>>>
segment('asgregorsamsaawokeonemorningfromuneasydreamshefoundhimselftr
inhisbedintoagiganticinsect')
['as', 'gregor', 'samsa', 'awoke', 'one', 'morning', 'from', 'uneasy'
'found', 'himself', 'transformed', 'in', 'his', 'bed', 'into', 'a', 'g
'insect']
>>>
segment('inaholeinthegroundtherelivedahobbitnotanastydirtywetholefill
endsofwormsandanoozysmellnoryetadrybaresandyholewithnothinginittosi
itwasahobbitholeandthatmeanscomfort')
['in', 'a', 'hole', 'in', 'the', 'ground', 'there', 'lived', 'a', 'hob
'nasty', 'dirty', 'wet', 'hole', 'filled', 'with', 'the', 'ends', 'of'
'an', 'oozy', 'smell', 'nor', 'yet', 'a', 'dry', 'bare', 'sandy', 'hol
'nothing', 'in', 'it', 'to', 'sitdown', 'on', 'or', 'to', 'eat', 'it',
'hobbit', 'hole', 'and', 'that', 'means', 'comfort']
>>>
segment('faroutintheunchartedbackwatersoftheunfashionableendofthewest
armofthegalaxyliesasmallunregardedyellowsun')
['far', 'out', 'in', 'the', 'uncharted', 'backwaters', 'of', 'the', 'u
'end', 'of', 'the', 'western', 'spiral', 'arm', 'of', 'the', 'galaxy',
'small', 'un', 'regarded', 'yellow', 'sun']
```

看到程序正确地对一些生僻的单词如“Samsa”和“oozy”进行分割，你可能会很高兴。对于“Samsa”在10000亿个单词中出现了42000次，“oozy”出现了13000次，你应该不会太惊讶。分割的整体结果看起来不错，但是有两个错误：'un'、'regarded'应该是一个单词，而'sitdown'应该是两个单词。尽管如此，分词的准确率是157/159=98.7%，还是不错的。

第一个错误是由于“unregarded”在我们的三十多万单词的词汇里（它在所有单词词汇中的位置是1005493，计数值是7557）。如果我们把它

放入词汇中，我们发现之前的分词是正确的。

```
>>>Pw['unregarded']=7557
>>>
segment（'faroutintheunchartedbackwatersoftheunfashionableendofthewest
armofthegalaxyliesasmallunregardedyellowsun'）
['far'，'out'，'in'，'the'，'uncharted'，'backwaters'，'of'，'the'，'u
```

这并没有证明我们解决了问题：我们需要放回所有其他的干扰单词，而不仅仅是我们需要的那个；而且我们需要返回所有的测试案例，确保增加其他的单词并没有混淆了任何其他的结果。

第二个错误是虽然“sit”和“down”是频繁词（概率分别为0.003%和0.04%），但是这两个单词的概率乘积刚好略小于“sitdown”这个单词本身的概率。根据二元计数，两个单词序列“sit down”的概率比其分别的概率乘积约大100倍。我们可以通过对二元建模来解决这个问题；也就是说，考虑每个单词的概率，给定前一个单词的概率是：

P(W1：n）=Πk=1：nP(Wk|Wk-1）

当然，整个二元单词表无法全部装载到内存中。如果我们只保留出现100000次以上的二元单词，结果就有250000多的入口项，这也无法装载到内存中。然而，我们可以通过Count(sit down)/Count(sit)来估计概率P(down|sit)。如果一个二元单词在表中不出现，我们就通过单元值来计算。在一个单词的前序单词给定的情况下，我们可以定义这个单词的条件概率cPw如下：

```
def cPw(word,prev):
"The conditional probability P(word|previous-word)."
try:
```

```
return P2w[prev+''+word]/float(Pw[prev]）
except KeyError:
return Pw(word)
P2w=Pdist(datafile（'count2w'），N）
```

（细心的人会注意到cPw不是概率分布，因为对于一个给定的先验单词，所有单词的概率之和会超过1。这种方法的技术名称是“愚蠢回退”(supid backoff)，但其实际效果很不错，因此我们不用担心。）我们现在来比较包含先验单词“to”、“sitdown”和“sit down”的概率：

```
>>>cPw（'sit'，'to'）*cPw（'down'，'sit'）/cPw（'sitdown'，'to'）
1698.0002330199263
```

我们发现“sit down”比“sitdown”的概率高1698倍，因为“sit down”是一个高频的二元项，而且“to sit”词频很高，而“to sitdown”词频不高。

这看起来振奋人心：让我们用二元模型来实现新版的分词。但要实现这种新版分词，我们还需要解决另外两个问题：

1.当给一个由n个单词构成的序列增加一个新的单词时，它调用Pwords来对这n+1个概率值进行乘法运算。但是分词时，原有的n个概率值已经做过一次运算处理了。将每个元素的运算结果也就是概率值保存起来，这样添加一个单词的时候就只需要另外执行一次运算，因此分词也将更为高效。

2.可能存在算术运算向下溢出问题。如果我们把Pwords应用于一个出现了61次“blah”之后的序列，我们将得到5.2×10^{-321}的概率，而如果再加上一个“blah”，其概率为0.0。我们能够表示的最小正浮点数是4.9×10^{-324}；任何小于该值的概率都置为0.0。为了避免向下溢出，最简单的解

决方案是对这些数值使用对数运算而非直接进行乘法运算。

我们将定义“segment2”（分词2），它和分词有三个方面的区别：第一，它使用了条件二元语言模型，cPw，而不是单元模型Pw。第二，函数特征是不同的。“segment2”除了需要把“文本”作为一个参数传递给函数之外，还需要将这段文本的一个单词也传递给函数。在句子的开始，前一个单词是句子的起始标记＜S＞；其返回值也不只是一个单词的序列，而是包含了两项内容：分词的概率和单词序列。我们返回概率，这样可以把它保存起来（通过记录的装饰器memo)，不需要重新计算；这解决了问题：（1）的效率低问题。合并函数接受四个输入：第一个单词、剩下的单词以及它们分别的概率。合并函数会把第一个单词添加到其他单词后面，并对所有的概率值进行乘法运算。但为了解决问题（2）[1]我们引入了第三个区别：不对概率值直接进行乘法运算而是对概率的对数执行加法运算。

以下“segment2”的代码：

```
from math import log10
@memo
def segment2(txt,prev='＜S＞')：
"Return(log P(words)，words)，where words is the best segmentation."
if not text:return 0.0，[]
candidates=[combine(log10(cw(first,prev))，first,segment2(rm,first))
for first,rem in splits(text)]
return max(candidates)
def combine(Pfirst,first，(Pem,rem))：
"Combine first and rem results into one(probability,words)pair."
return Pfirst+Prem，[first]+rem
```

segment2执行了O(nL)次的递归调用，每次调用需要考虑O(L)次的切分，因此整个算法的复杂度是$O(nL^2)$。实际上，这就是Viterbi算法，memo隐含地创建Viterbi表。

segment2正确地对“sit down”例子进行分割，第一个版本中能正确切分的例子，在这个版本中也仍然是正确的。但这两个版本对“unregarded”的分词都不正确。

我们能否在性能上有所提升？很有可能。我们可以对未知单词创建一个更精确的模型；可以组合更多的数据，或者在单元数据或者二元数据中保留更多的入口项，或者添加三元数据。

[1] 乘法运算的向下溢出问题。

密码

我们所面临的第二个挑战是对加密的消息进行解密。我们首先一起来看看替换加密，即使用一个字母替换另一个字母。对其中替换关系的描述称为密钥，可以通过26个字母的字符串来表示它：第一个字母替换“a”，第二个字母替换“b”等。以下是通过替换密码密钥对消息进行加密(Pthon的库函数maketrans和translate可以实现大部分功能）：

```
def encode(msg,key):
"Encode a message with a substitution cipher."
return msg.translate(string.maketrans(ul(alphabet), ul(key)))
def ul(text): return text.upper()+text.lower()
alphabet='abcdefghijklmnopqrstuvwxyz'
```

可能所有代码中最简单的就是移位加密，它是一种替换加密方法，在这种加密方法中，消息中的每个字母都由其后面的第n个字母进行替换。如果n=1，那么“b”替换“a”，“c”替换“b”……直到“a”替换“z”。移位加密也被称为凯撒(Cesar)加密法；它们在公元前50多年很先进。变换(sift)函数使用移位加密的方法进行编码：

```
def shift(msg,n=13):
"Encode a message with a shift(Caesar)cipher."
return encode(msg,alphabet[n: ]+alphabet[: n])
我们如下使用该函数：
>>>shift('Listen,do you want to know a secret?')
'Yvfgra,qb lbh jnag gb xabj n frperg?'
>>>shift('HAL 9000 xyz', 1)
'IBM 9000 yza'
```

在不知道密钥的情况下进行解密，我们遵循和分段相同的方法：定义一个模型（我们将继续采用单元单词概率方式），枚举候选项并选择最有可能的。由于只需要考虑26种候选移位，所以我们可以尝试全部候选移位。

为了实现这一点，我们定义函数logPwords，它和函数Pwords相似，但是它返回的是概率的自然对数值，其输入可以是一个很长的字符串，也可以是一个单词序列：

```
def logPwords(words):
"The Naive Bayes probability of a string or sequence of words."
if isinstance(words,str): words=allwords(words)
return sum(log10(P(w)) for w in words)
def allwords(text):
"Return a list of alphabetic words in text,lowercase."
return re.findall（'[a-z]+', text.lower（））
现在我们可以枚举所有候选项进行解码并挑选最有可能的：
def decode_shift(msg):
"Find the best decoding of a message encoded with a shift cipher."
candidates=[shift(msg,n) for n in range(len(alphabet))]
return max(candidates,key=logPwords)
我们可以通过测试来检查这种方式是否能够工作：
＞＞＞decode_shift（'Yvfgra,qb lbh jnag gb xabj n frperg?'）
'Listen,do you want to know a secret?'
```

这一切都太简单了。为了查看为什么简单，观察以下26个候选项，以及它们的概率对数值：

```
Yvfgra,qb lbh jnag gb xabj n frperg?-84
Zwghsb,rc mci kobh hc ybck o gsqfsh?-83
Axhitc,sd ndj lpci id zcdl p htrgti?-83
Byijud,te oek mqdj je adem q iushuj?-77
Czjkve,uf pfl nrek kf befn r jvtivk?-85
Daklwf,vg qgm osfl lg cfgo s kwujwl?-91
Eblmxg,wh rhn ptgm mh dghp t lxvkxm?-84
Fcmnyh,xi sio quhn ni ehiq u mywlyn?-84
Gdnozi,yj tjp rvio oj fijr v nzxmzo?-86
```

```
Heopaj,zk ukq swjp pk gjks w oaynap?-93
Ifpqbk,al vlr txkq ql hklt x pbzobq?-84
Jgqrcl,bm wms uylr rm ilmu y qcapcr?-76
Khrsdm,cn xnt vzms sn jmnv z rdbqds?-92
Listen,do you want to know a secret?-25
MjtufO'ep zpv xbou up lopx b tfdsfu?-89
Nkuvgp,fq aqw ycpv vq mpqy c ugetgv?-87
Olvwhq,gr brx zdqw wr nqrz d vhfuhw?-85
Pmwxir,hs csy aerx xs orsa e wigvix?-77
Qnxyjs,it dtz bfsy yt pstb f xjhwjy?-83
Royzkt,ju eua cgtz zu qtuc g ykixkz?-85
Spzalu,kv fvb dhua av ruvd h zljyla?-85
Tqabmv,lw gwc eivb bw svwe i amkzmb?-84
Urbcnw,mx hxd fjwc cx twxf j bnlanc?-92
Vscdox,ny iye gkxd dy uxyg k combod?-84
Wtdepy,oz jzf hlye ez vyzh l dpncpe?-91
Xuefqz,pa kag imzf fa wzai m eqodqf?-83
```

当你扫描列表时，刚好一条线很明显，类似英语；而Pwords和我们的直觉一致，假设该行的概率的对数值是-25（即10^{-25}），它的可能性是任何其他候选项的10^{50}倍多。

代码编写者通过消除单词之间的标点符号、空格和大小写区别，使得解码者的工作变得更加困难。采用这种方式，解码者无法从简短的单词如“I”、“a”和“the”中得到任何线索，也无法猜测省略符号（'）之后的字符应该是“s”还是“t”。函数shift2是一种加密方法，它删除了非字符的数据，把所有的字符都转换成小写，然后再应用变换加密方法：

```
def shift2(mg,n=13）:
"Encode with a shift(Caesar)cipher,yielding only letters[a-z]."
return shift(just_letters(msg)，n）
def just_letters(text):
"Lowercase text and remove all characters except[a-z]."
return re.sub（'[^a-z]'，''，text.lower（）)
```

以下是破译这段代码的一种方式，通过枚举每个候选项，对每个候

选项分段，并选择概率值最高的那个分段：

```
def decode_shift2(mg):
"Decode a message encoded with a shift cipher,with no spaces."
candidates=[segment2(sift(msg,n）) for n in range(len(alphabet))]
p,words=max(candidates)
return''.join(words)
```

我们一起来看它是如何工作的：

```
>>>shift2（'Listen,do you want to know a secret?')
'yvfgraqblbhjnaggbxabjnfrperg'
>>>decode_shift2（'yvfgraqblbhjnaggbxabjnfrperg')
'listen do you want to know a secret'
>>>decode_shift2(sift2（'Rosebud'))
'rosebud'
>>>decode_shift2(sift2（"Is it safe?"))
'is it safe'
>>>decode_shift2(sift2（"What's the frequency,Kenneth?"))
'whats the frequency kenneth'
>>>msg='General Kenobi:Years agO'you served my father in the
Clone
Wars; now he begs you to help him in his struggle against the
Empire.'
>>>decode_shift2(sift2(mg))
'general kenobi years ago you served my father in the clone wars
now he
begs you to help him in his struggle against the empire'
```

这还是太简单了。我们一起来看看通用的密码替换方案，在这种方案中，任何字母都可以被其他字母替换。现在我们无法继续枚举所有的概率，因为总共有26！（约4×10^{26}）种密钥，而不再仅仅是26种。Simon Singh（编码鼻祖）编写的《The Code Book》提供了五种策略（我们自己在后面增加了第六种）来破解密码：

1.字母一元模型频率。匹配消息中的常用字母到英语中的常用字母

（如“e”），匹配不常用字母到不常用字母（如“z”）。

2.双字字母分析。加密消息中的双字字母，在消息解密后还是双字字母。考虑最生僻和最常用的双字字母。

3.找常见的单词如“the”、“and”和“of”。单字母单词通常是“a”或“I”。

4.可能的话，获取一个由你处理的消息类型所组成的词频表。如军事消息使用军事术语等。

5.猜测一个单词或者短语。例如，如果你觉得消息可能包含“your faithful servant”，就大胆猜测它。

6.使用单词模式。例如，加密单词“abbccddedf”很可能表示“bookkeeper”，因为在词典库中没有其他单词满足这种模式。

对于单词间不包含空格的消息，策略3和策略6都不适合。策略1和策略2每种只包含26种概率，而且似乎目标受众是有一定记忆和计算能力的分析人员，而不是一个计算机程序。策略4和策略5是为了某个特殊用途，而不是通用的解码方式。看起来我们只能依赖于自己提出的策略6。但是我们知道应该如何实现该方法。

第一，定义一个概率模型：我们可以采用处理移位加密时所用的方式来评估候选项：对文本分段，计算单词的概率。但是考虑该方法的第二个步骤，刚开始几个（或者几千个）候选项可能是很差的候选。在开始探索时，我们没有任何类似于单词的东西，因此在刚开始就对文本分段的意义不大。然而，我们可能（或者只是碰巧）对一行的几个字母进行解码，生成有意义的结果。因此，我们使用N元文法模型而不是单词

来构建语言模型。我们应该采用字母二元文法模型？还是三元文法模型？抑或五元文法模型？我选择字母三元文法模型是因为它是能够表示常用词（策略3）的最短的文法模型。我通过一个单词二元字母数据文件（不能只看词汇文件，因为需要考虑单词边际间的字母三元文法模型），对三元字母（删去了空格和标点符号）进行计数，生成数据文件count_3l。它总共包含26^3=17576个三元字母组合。以下是出现频率最高和最低的10个三元字母组合：

```
the 2.763%fzq 0.0000004%
ing 1.471%jvq 0.0000004%
and 1.462%jnq 0.0000004%
ion 1.343%zqh 0.0000004%
tio 1.101%jqx 0.0000003%
ent 1.074%jwq 0.0000003%
for 0.884%jqy 0.0000003%
ati 0.852%zqy 0.0000003%
ter 0.728%jzq 0.0000002%
ate 0.672%zgq 0.0000002%
```

三元字母组合的概率计算如下：

```
def logP3letters(text):
"The log-probability of text using a letter 3-gram model."
return sum(log10（P3l(g)）for g in ngrams(text，3）)
P3l=Pdist(datafile（'count_3l'）)
P2l=Pdist(datafile（'count_2l'）)##We'll need it later
```

第二，枚举候选项：我们无法考虑全部的4×10^{26}种可能的密钥，而似乎没有什么方法可以像文本分段那样，系统性地消除非最优候选项。这种情况下，需要本地搜索策略，如“爬山算法”(hll climbing)。假设你想要到达最高峰，但是没有地图。通过爬山算法的策略，你可以从随机

位置x开始，向相邻位置走一步。如果该位置比之前位置高，继续从那里开始“爬山”；如果不是，则考虑x的另一个相邻位置。当然，如果你从地球上随机一个位置开始步行上山，你可能无法到达珠穆朗玛峰的顶部。更有可能的是，你到达了某个地方的小山头，或者困在一个平原里到处走。因此，我们为爬山算法增加了随机次数的重试。在完成这些步骤后，我们开始从一个新的随机位置从头开始。

以下是通用的爬山算法。它包含起始位置x，我们正试着优化的函数f，生成一个位置的相邻位置的函数neighbors，以及最多采取的步骤。（如果变量调试是真实的，它会输出最佳的位置x及其分值）。

```
def hillclimb(x,f,neighbors,steps=10000）:
"Search for an x that miximizes f(x)，considering neighbors(x)."
fx=f(x)
neighborhood=iter(neighbors(x)）
for i in range(steps):
x2=neighborhood.next（）
fx2=f(x2）
if fx2＞=fx:
x,fx=x2，fx2
neighborhood=iter(neighbors(x)）
if debugging:print'hillclimb: '，x,int(fx)
return x
debugging=False
```

为了使用爬山算法来解码，我们需要指定参数。我们将要搜索的位置是纯文本（解码后的）消息。我们将试着最大化三元字母组词频，因此，设置函数f=logP3letters。从根据随机的密钥解码生成的x位置开始，做随机次数的重试，但是当从每次重试中收集候选项时，我们会根据segment2，选择最佳的一个候选项，而不是根据函数logP3letters：

```
def decode_subst(msg,steps=4000，restarts=20）:
"Decode a substitution cipher with random restart hillclimbing."
msg=cat(allwords(msg)）
candidates=[hillclimb(encode(msg,key=cat(shuffled(alphabet)）)，
logP3letters,neighboring_msgs,steps)
for_in range(restarts)]
p,words=max(segment2(c)for c in candidates)
return''.join(words)
def shuffled(seq):
"Return a randomly shuffled copy of the input sequence."
seq=list(seq)
random.shuffle(seq)
return seq
cat=''.join
```

现在，我们需要定义相邻消息函数neighboring_msgs，它生成下一步要尝试的消息的解码。我们首先尝试“修复”不可能的二元字母组。举个例子，最不频繁出现的二元字母组合“jq”的概率是0.0001%，它比最频繁出现的二元字母组合“in”和“th”的概率低5万倍。因此，如果我们在消息中看到“jq”，我们试着把“j”和其他每一个字母进行交换，同样也交换“q”。如果一个交换生成更频繁的二元字母组，那么我们生成由这次交换得到的消息。在遍历20次最不可能出现的二元字母组“修复”后，我们考虑随机交换：

```
def neighboring_msgs(msg):
"Generate nearby keys,hopefully better ones."
def swap(a,b）: return msg.translate(string.maketrans(a+b,b+a）)
for bigram in heapq.nsmallest（20，set(ngrams(msg，2）)，P2l）:
b1，b2=bigram
for c in alphabet:
if b1==b2:
ifP2l(c+c）＞P2l(bigram): yieldswap(c,b1）
else:
if P2l(c+b2）＞P2l(bigram): yield swap(c,b1）
if P2l(b1+c）＞P2l(bigram): yield swap(c,b2）
while True:
yield swap(random.choice(alphabet)，random.choice(alphabet)）
```

我们一起来看看它的效果如何。我们将从Robert Raynard的书《Secret Code Breaker》(Sith和Daniel；见http：//secretcodebreaker.com)中抽取一些密码来尝试这种策略。首先是一则“热身”的加密消息：

```
＞＞＞msg='DSDRO XFIJV DIYSB ANQAL TAIMX VBDMB GASSA QRTRT CGGXJ MMTQC IPJSB AQPDR
SDIMS DUAMB CQCMS AQDRS DMRJN SBAGC IYTCY ASBCS MQXKS CICGX RSRCQ ACOGA SJPAS
AQHDI ASBAK GCDIS AWSJN CMDKB AQHAR RCYAE'
＞＞＞decode_subst(msg)
'it is by knowing the frequency which letters usually occur and other distinctive
characteristics of the language that crypt analysts are able to determine the
plain text of a cipher message j'
```

解码后的消息，除了“crypt analyst”本应该是一个单词，其他都是正确的。（这个词不在我们的词库中，但是它在1300亿的单词库中）。注意最后一个字符（加密文本的“E”）是为了使所有分块都是五个字母一组而增加的。

以下是来自第一次世界大战时德国间谍Baron August Schluga的真正的密文：

```
＞＞＞msg='NKDIF SERLJ MIBFK FKDLV NQIBR HLCJU KFTFL KSTEN YQNDQ NTTEB TTENM QLJFS
NOSUM MLQTL CTENC QNKRE BTTBR HKLQT ELCBQ QBSFS KLTML SSFAI NLKBR RLUKT LCJUK
FTFLK FKSUC CFRFN KRYXB'
＞＞＞decode_subst(msg)
'english complaining over lack of munitions they regret that the promised support of
the french attack north of arras is not possible on account of munition insufficiency wa
```

以下是在1992年，美国国家安全委员会(KB)向美国前中央情报局(CA)主席Aldrich Ames发送的密文，Aldrich Ames在1994年被判为间谍：

```
>>>msg='CNLGV QVELH WTTAI LEHOT WEQVP CEBTQ FJNPP EDMFM LFCYF SQFSP NDHQF OEUTN
PPTPP CTDQN IFSQD TWHTN HHLFJ OLFSD HQFED HEGNQ TWVNQ HTNHH LFJWE BBITS PTHDT
XQQFO EUTYF SLFJE DEFDN IFSQG NLNGN PCTTQ EDOED FGQFI TLXNI'
>>>decode_subst(msg)
'march third week bridge with smile to pass info from you to us and to give
assessment about new dead drop ground to indicate what dead drop will be used next
to give your opinion about caracas meeting in october xab'
```

以下是在1943年德国的U-Boat命令的密文被截获并且被解码，从而挽救了联合战舰护航队：

```
msg='WLJIU JYBRK PWFPF IJQSK PWRSS WEPTM MJRBS BJIRA BASPP IHBGP RWMWQ SOPSV PPIMJ
BISUF WIFOT HWBIS WBIQW FBJRB GPILP PXLPM SAJQQ PMJQS RJASW LSBLW GBHMJ
QSWIL PXWOL'
>>>decode_subst(msg)
'a cony ov is headed northeast take up positions fifteen miles apart between point
yd and bu maintain radio silence except for reports of tactical importance x abc'
```

解码后的消息混淆了“y”和“x”。分析人员会意识到“conyov”应该是单词“convoy”，因此“point yd”应该是“point vd”。我们的程序从未考虑到这种可能性，因为正确文本的三元字母组合概率比这里结果所显示的要低。通过发明一种更好的打分函数，它不会陷于局部最优中，我们可能

可以解决这个问题。或者我们可以增加第二层次的爬山搜索：通过第一次搜索生成的候选项，以segment2作为打分函数来简单搜索。我们将把它作为练习留给读者。

拼写纠正

我们的最后一个任务是拼写纠正：给定一个输入的单词w，确定单词c是其最可能要表达的单词。举个例子，如果w是“acomodation”，c应该是“accommodation”。（如果w是“the”，那么c也应该是“the”。）

根据标准的方法，我们想要选择c，使得其条件概率P(c|w）值最大。但是如何定义该概率却不简单。假设w=“thew”，那么c的一个候选项是“the”——它是最常见的单词，我们可以想象是打字员的手指输入“e”后不小心按到“w”。另一个候选项是“thaw”——一个较常见的单词（虽然出现概率比“the”低3万倍），而且用一个原因取代另一个很常见。其他的候选项包括“thew”本身（表示肌肉或者肌腱的模糊术语），“threw”和一种姓氏“Thwe”。我们应该选择哪一种？看起来我们合并了两种因素：c本身的概率是多大？w是c的输入错误，或者是拼读错误，或者是其他某种形式的误拼，它们的概率有多大？有人可能会认为我们将通过某种特殊(a hoc)方式来结合这些因素，但是实际上存在一个数学公式——贝叶斯理论——精确地告诉我们如何组合这些因素来找到最佳的候选项：

argmaxc P(c|w）=argmaxc P(w|c）P(c)

这里，P(c)表示c是期望的结果单词的概率，称为语言模型；P(w|c）表示用户期望获取c会输入w的概率，称为错误模型或者噪音渠道模型。（其想法是用户希望输入c，但是由于一些噪音或静态数据，改成了输

入w。）遗憾的是，从我们拥有的语料数据来看，对该模型进行估计并不容易——语料库没有提供信息表示哪些单词是其他单词的拼写错误。

我们可以通过更多的数据来解决这个问题：一个错误拼写列表。Roger Mitton在http：//www.dcs.bbk.ac.uk/～ROGER/corpora.html维护了一个约1万条（c,w）词对的数据语料库。但是我们不能只是从该数据中查找P(w=thew|c=thaw)；若只有4万条例子，那么看到相同的词对的概率很低。当数据很稀疏时，需要对它进行范化。可以通过忽略相同的字母、“th”和“w”来进行范化，计算P(w=e|c=a），即期望输入字母“a”却输入“e”的概率。由于存在如“consistent/consistant”和“inseparable/inseperable”这样的错误，把“a”误拼写为“e”是最常见的一种单词误拼。

在以下表中，我们考虑当w=thew时，c的五个候选项。其中一个是thew本身，另外四个表示我们将要考虑的四种单步编辑类型：1）删除字母“thew”中的“w”，生成“the”。2）插入“r”，得到“threw”。以上两种编辑都是考虑编辑其前一个字母。3）用“a”替换“e”，如之前所述。4）对两个相邻字母交换位置，把“ew”交换成“we”。我们称这些单步编辑为编辑距离1；需要两步编辑的候选项是编辑距离2。下表显示单词w和c，编辑w|c，概率P(w|c），概率P(c)，以及二者概率的乘积（为了可读性进行了扩展）。

w	c	w | c	P(w | c)	P(c)	10^9 P(w | c) P(c)
thew	the	ew | e	0.000007	0.02 144.	144.
thew	thew		0.95	0.00000009	90.
thew	thaw	e | a	0.001	0.0000007	0.7
thew	threw	h | hr	0.000008	0.000004	0.03
thew	thwe	ew | we	0.000003	0.00000004	0.0001

从表中可以看出，“the”是最可能的纠正结果。P(c)可以和Pw一起计算。对于P(w|c），我们需要创建一个新的函数Pedit，它给出了一步编辑操作的概率，从拼写错误的语料库中估计出来。举个例子，可以计算Pedit（'ew|e'）=0.000007。更复杂编辑定义成单步编辑的连接。例如，为了把“hallow”转换成“hello”，我们把a|e编辑操作和ow|o编辑操作连接起来，因此整个编辑被称为a|e+ow|o（或者ow|o+a|e，在这种情况下（但不总是如此）是相同的）。复杂编辑的概率等于其各个子部分的乘积。

存在一个问题：空编辑Pedit（''）的概率有多大？也就是说，假设期望的单词是c，人们真的输入c而不是其中某种错误的编辑？这依赖于打字员的熟练程度以及是否有检查。很随意地，我假定每20个单词出现一次拼写错误。值得注意的是，如果我假定每50个单词才出现一次错误，那么对于w=“thew”，P(w|c）的概率为0.98.，而不是0.95，而且“thew”将是最可能的结果。

最后，我们可以展示代码了。该代码包含两个高层函数correct和corrections,correct函数返回一个单词的最佳纠正，corrections函数把correct应用到文本中的每个单词，对其邻近字符不做改动。候选项是所有可能的编辑，而最佳候选项是有最高的P(w|c）P(c)分值的候选项。

```
def corrections(text):
"Spell-correct all words in text."
return re.sub（'[a-zA-Z]+', lambda m:correct(m.group（0））, text)
def correct(w):
"Return the word that is the most likely spell correction of
w."candidates=edits(w).items（）
c,edit=max(candidates,key=lambda(c,e）: Pedit(e)*Pw(c)）
return c
P(w|c）是通过函数Pedit来计算的:
def Pedit(edit):
"The probability of an edit; can be''or'a|b'or'a|b+c|d'."
if edit=='': return（1.-p_spell_error)
return p_spell_error*product(P1edit(e)for e in edit.split（'+'））
p_spell_error=1./20.
P1edit=Pdist(datafile（'count1edit'））##Probabilities of single
edits
```

候选项是通过编辑生成的，即传递一个单词，返回一个{单词：编辑}对的词典dict表示可能的纠正。通常来说，可能对于一个纠正存在多种编辑。（例如，我们通过在“e”或者“l”后面插入“l”后，都可以把“tel”转为“tell”。）我们选择有最高概率的编辑。edits是我们目前为止最复杂的函数。一部分原因是该函数本质上就很复杂，生成4种编辑很复杂。但是一部分是因为我们采取了一些措施使得该函数变得高效。（性能较低但是易于读取的一个版本在http：//norvig.com/spell-correct.html。）如果我们考虑所有的编辑，一个单词如“accommodations”可以生成233166个候选项。但是只有11种候选项可以在词汇库中找到。因此，编辑是通过预计算词汇库中所有单词的所有前缀才能工作。然后，它递归调用edits，把单词分成头部和尾部（代码中的hd和tl)，并保证头部总是在前缀列表中。通过把结果添加到词典库dict中进行收集：

```
def edits(word,d=2）:
"Return a dict of{correct:edit}pairs within d edits of word."
results={}
def editsR(hd,tl,d,edits):
def ed(L,R）: return edits+[R+'|'+L]
C=hd+tl
if C in Pw:
e='+'.join(edits)
if C not in results:results[C]=e
else:results[C]=max(results[C]，e,key=Pedit)
if d＜=0: return
extensions=[hd+c for c in alphabet if hd+c in PREFIXES]
p=(h[-1]if hd else'＜'）##previous character
##Insertion
for h in extensions:
editsR(h,tl,d-1，ed(p+h[-1]，p）)
if not tl:return
##Deletion
editsR(hd,tl[1: ]，d-1，ed(p,p+tl[0]）)
for h in extensions:
if h[-1]==tl[0]: ##Match
editsR(h,tl[1: ]，d,edits)
else: ##Replacement
editsR(h,tl[1: ]，d-1，ed(h[-1]，tl[0]）)
##Transpose
if len(tl)＞=2 and tl[0]！=tl[1]and hd+tl[1]in PREFIXES:
editsR(hd+tl[1]，tl[0]+tl[2: ]，d-1，
ed(tl[1]+tl[0]，tl[0: 2]）)
##Body of edits:
editsR（''，word,d，[]）
return results
PREFIXES=set(w[: i]for w in Pw for i in range(len(w)+1）)
```

以下是使用edits函数的一个例子：

```
＞＞＞edits（'adiabatic'，2）
{'adiabatic': ''，'diabetic': '＜a|＜+a|e'，'diabatic': '＜a|＜'}
```

以下是正在工作的拼写纠正器：

```
＞＞＞correct（'vokabulary'）
'vocabulary'
＞＞＞correct（'embracable'）
'embraceable'
```

```
>>>corrections('thiss is a teyst of acommodations for korrections
of mispellings of particuler wurds.')
'this is a test of acommodations for corrections of mispellings
of particular words.'
```

15个单词中有13个处理结果正确，但是“acommodations”和“mispellings”处理结果错误。为什么呢？遗憾的是，网络上充满了很多错误的拼写。错误的单词“mispellings”在语料库中出现了18543次，而正确的单词“misspellings”只出现了50万次，这其中的差距不足以带来用一次编辑替换无编辑。“thew”出现了96759次，我怀疑其中绝大多数也是由于拼写错误。

有很多种方式可以改进这个拼写程序。首先，我们可以通过上下文的其他单词来纠正这个单词，因此“they're”本身是正确的，但是当出现在“in they're words”时，就会被纠正为“their”。一个单词二元模型或者三元模型就可以实现它。

我们真的需要清洗语料库中的拼写错误。看看以下这些不同的“misspellings”：

```
misspellings 432354
mispellings 18543
misspelings 10148
mispelings 3937
```

语料库中使用的misspellings一词，拼写错误的概率达到了7%。可以有三种方式来解决这个问题。第一种，我们可以获得一个字典单词列表，并且只对字典中的单词纠正错误。但是字典没有列出最新发明的单

词和名字。（一种折中方案是把单词变成小写放到字典词汇中，但是允许全部大写的单词不在字典库中。）第二种，我们可以获得一个已经仔细校验过的语料库，可能是来自于高品质的出版社的书籍和期刊。第三种，我们可以纠正语料库中的错误。在使用语料库进行拼写检查之前，先对语料库拼写检查，这看起来就像是“循环论证”(crcular reasoning)，但是可以实现。对于这个应用，我们把编辑距离很小的单词聚在一起。针对每组相似单词，我们检查是否有某个单词的出现频率要远远高于其他单词。如果存在这个单词，我们将检查二元模型（或者三元模型）计数来观察这两个单词和相邻的单词是否有相似的分布。例如，以下是“mispellings”和“misspellings”的四个二元文法模型计数：

```
mispellings allowed 99
misspellings allowed 2410
mispellings as 50
misspellings as 749
mispellings for 122
misspellings for 11600
mispellings of 7360
misspellings of 16943
```

这两个单词共享很多相同的二元文法相邻的单词，对于这些单词组合中，总是“misspellings”的出现频率更高，因此很有可能“mispellings”是拼写错误。初级测试表示这种方法效果不错——但是存在一个问题：它需要CPU计算好几百个小时。因此，这种方法适合于在一个工作组上执行计算，而不适合于单机。

这种数据驱动的方法与更传统的软件开发过程相比如何？对于后

者，程序员在编码中制定了一些显示规则。为了回答这个问题，我们一起来看ht：//Dig项目的拼写纠正代码，Dig是一款优秀的开源企业内网搜索引擎。给定一个单词，ht：//Dig项目的metaphone算法可以生成一个关键字来表示这个词的发声。举个例子，“tough”和“tuff”都映射到关键字“TF”，因此可以作为互相拼写错误的候选。以下是对于字母“G”的metaphone算法的代码：

```
case'G':
/*
*F if in-GH and not B--GH,D--GH,
*-H--GH，-H---GH else dropped if
*-GNED，-GN，-DGE-，-DGI-，-DGY-
*else J if in-GE-，-GI-，-GY-and
*not GG else K
*/
if（（*（n+1）！='G'||vowel（*（n+2）））＆＆
（*（n+1）！='N'||（*（n+1）＆＆
（*（n+2）！='E'||
*（n+3）！='D'）））＆＆
（*（n-1）！='D'||！frontv（*（n+1））））
if(frontv（*（n+1））＆＆*（n+2）！='G'）
key<<'J';
else
key<<'K';
else if（*（n+1）=='H'＆＆！noghf（*（n-3））＆＆
*（n-4）！='H'）
key<<'F';
break;
```

这块代码正确地把“TOUGH”中的“GH”映射为“F”，但是不会把“BOUGH”中的“GH”映射为“F”。但是这些规则是否正确地捕获了所有的情况？“OUGHT”呢？或者“COUGH”和“HICCOUGH”？我们可以编写测试样例来验证代码的每个分支，但是即使如此，我们还是无法知道

哪些方面没有覆盖全。当引入新的单词如“iPhone”，结果会如何呢？显然，手工制定的规则很难扩展和维护。数据驱动方法的很大优势是在数据中包含了很多信息，可以简单地通过增加更多的数据来增加新的知识。另一个优点是，虽然数据是海量的，但代码却很简洁——函数correct只包含50行左右的代码，而ht：//Dig的拼写代码超过1500行。正如伟大的Bill Gate所言：“通过代码行数来衡量编程进展正如通过重量来衡量飞机建造过程。”Gates认为代码行数是一种负担，而不是财富。概率性的数据驱动方法是敏捷开发的终极。

该算法的另一个问题是可移植性。如果我们想要一个拉脱维亚语(Ltvian)的拼写纠正器，该英语metaphone规则几乎毫无用处。把数据驱动的纠正算法移植到另一种语言上，我们需要的是一个大的拉脱维亚语的语料库，而不需要改动代码。

其他任务

以下是通过概率性语言模型处理的一些其他任务。

语言识别

存在Web协议来声明网页的编码语言。实际上至少存在两种协议，一种在HTML，另一种在HTTP。但是有时这两种协议不一致，它们都不表示真实的信息，因此搜索引擎在搜集了一些已知语言的样本页面后，通常通过基于实际页面内容对页面进行分类。任务是写一个这样的分类器。该技术的前沿水平是语言识别的准确率超过99%。

垃圾邮件检测和其他分类任务

据估计，每天发送的垃圾邮件有1000亿封。给定垃圾邮件和非垃圾邮件两个语料库，任务是对未来的消息进行分类。最佳的垃圾邮件分类器包含N元单词模型（包含“10000000.00 will be released”和“our country Nigeria”的消息很可能是垃圾邮件）以及N元字符模型（“v1agra”可能是垃圾邮件）等特征。该任务的前沿技术是通过垃圾邮件拦截方式，其准确率超过99%。一旦你可以对文档分类为垃圾邮件/非垃圾邮件，那么你就可以很容易地进行其他方式的分类，比如紧迫/非紧迫邮件消息，或者对新闻文章的分类，如政治/商业/体育/其他，或者产品反馈的分类如“喜欢/一般/不喜欢”。

作者识别（文体学Stylometry)

语言模型已经被用于识别《Federalist Papers》、莎士比亚的诗和

《圣经》的有争议的作者。相似的技术还被用于追踪恐怖组织，在刑法中，用于识别和找到罪犯。该领域还不太成熟，我们还不太确定什么是最佳实践，或者期望概率是多少，虽然2004年竞赛的胜出者的准确率达到71%。竞赛的最佳实践在语言上很简单，但是在统计上非常复杂。

记录文档重组和DNA序列化

在Vernor Vinge的科幻小说《Rainbows End》(Tr Books出版），Librareome项目通过把所有书籍扔到树碎纸机、对碎片拍照和使用计算机算法重新组合这些图片的方式数字记录整个图书馆的信息。在实际生活中，德国政府的E-Puzzler项目正在重新构建4500万页面的记录文档，它们被民主德国秘密警察Stasis所破坏。这两个项目都依赖复杂的计算机视觉技术。但是一旦把图像转化为字符，即可使用语言模型和爬山算法来重新组合那些碎片。可以采用类似的技术来解读生命之语：人类基于组项目使用了称为shotgun序列化技术来重新组合DNA碎片。因此，所谓的“下一代序列化”把更多的负担从Web实验室转移到大规模的并行重组算法。

机器翻译

G公司的N元语料是由机器翻译组的研究人员创建的。把外语(f)翻译成英语(e)和纠正拼写错误的单词很相似。最佳的英语翻译建模如下：

best=argmaxe P(e|f）=argmaxe P(f|e）P(e)

其中，P(e)是英语的语言模型，通过N元语法模型数据进行估计，而P(f|e）是翻译模型，通过双语语料库进行学习：在双语语料库中，一对

文档被标记为相互的翻译。虽然高端系统利用了语言学特征，包括很多句子的一部分词和语义解析，但是结果发现翻译需要的绝大部分知识来自于n元文法模型数据。

讨论和结论

我们显示了软件开发方法的力量，它使用大量的数据来解决不确定环境中提出的不确定问题。本章中探讨的是语言数据，但是很多相同的经验也适合于其他数据。

在我们所探索的样例中，程序简洁明了，这是因为使用的概率模型简单。这些样例模型忽略了人类所知道的很多知识——显然，对“choosespain.com”进行分段，我们需要了解很多关于旅游业如何工作以及其他因素的专业知识，但是奇怪的结果在于程序不需要显式表示所有的知识；它隐式地从N元语法模型中获取知识，它表示其他人选择了探讨的方面。在过去，概率模型更复杂，因为它们依赖于更少的数据。

需要强调的是，当缺失数据时的统计复杂的平滑策略。既然有非常大的语料库，那么我们可以使用类似“愚蠢回退”的方法，而不再过于担心平滑模型。

我们在本章研究的程序中的绝大多数复杂性是由于搜索策略。我们看到了三类搜索策略：

穷举式(Ehaustive)

对于移位密码，只有26个候选项，我们可以全部一一测试它们。

保证式(Garanteed)

对于分段策略，存在2^n种候选方式，但是很多方式可以被证明是非最优的（在独立假设前提下），不需要对它们进行检查。

启发式(Huristic)

对于全部的替换密码，我们无法保证已经发现了最佳的候选，但是我们可以搜索一个代表子集，使我们可以有较大的机会找到最佳候选。对于很多问题，其绝大部分工作是选择正确的函数进行优化，理解搜索空间的拓扑，发现好的顺序来枚举相邻节点。

如果我们是基于大量的数据构建模型，那么就需要获取“在各种情况下”(i the wild)可能的数据。N元文法计数具备这个属性：我们可以很容易地从Web中收集1万亿条自然文本数据。然而，标签化的拼写纠正不是自然产生的，因此我们在这些数据中只找到了4万个。两种最成功的自然语言应用——机器翻译和语言识别——可以得到大量的语料样例，这绝非偶然。相反，句子的语义解析任务依然大部分还未实现，一部分是由于在解析后的句子中不存在自然存在的大量的语料库。

值得一提的是，我们的概率性数据驱动模型方法——最大化了所有候选项的概率——是合理的数据驱动方法的一个特例——最大化所有候选项的期望效益。举个例子，合理性的拼写纠正程序需要知道存在一些禁忌语，而当用户并不想使用这些禁忌语时，提示这些禁忌语会给用户带来难堪，其负面影响比拼写一个错误单词要严重得多。合理性的程序会考虑一个单词正确或错误的概率，以及提示每个单词所带来的正面或负面价值。不确定问题需要在校验和测试上有很好的纪律约束。为了评估一个不确定问题的解决方案，人们需要把数据分为三个集合：1）训练集合，用于构建概率模型。2）校验集合，开发人员用于评估一些不

同的方法，观察哪一种方法打分更高，从新的算法中获取思想。3）测试集合，用于在开发介素时准确地判断该算法对于新的、未访问的数据效果如何。当使用完一次测试集合后，理想情况是抛弃该集合，正如一个老师不能给一个班级的学生进行两次相同的测试。但是在实践中，数据很昂贵，而且在通过支付方式获取新数据和通过某种不会导致模型过度拟合数据的方式重用旧数据之间存在一个折中。注意对独立测试集合的需要对于未知问题本身是固有的，而不是和选择的解决方案类型有关——即使你决定通过特定规则(a hoc rule)而不是概率模型的方式来解决问题，你都应该使用合适的测试方法。

总之，由于网络上可以获取更多的数据，而且随着计算能力的提高，我相信概率性数据驱动方法将会成为解决未知领域的复杂问题的主要策略。

致谢

感谢Darius Bacon、Thorsten Brants、Andy Golding、Mark Paskin、Franco Salvetti和Casey Whitelaw提供评论、纠正和代码相关的建议。

第15章　数据中的生命：DNA漫谈

MattWood和Ben Blackburne

DNA不仅仅是组成生物体的积木，它还是生物体化学组成的数据库。这个数据库由数百万年来芸芸众生的演化所设计实现，它简单、无模式、可容错。在过去的20年中，随着基因手段在现代生物医学研究中所占的比重增大，生物学家的研究重点也开始从单个基因转移到整个基因组上。最近几年来，生物学家一直在研究如何在DNA上读写数据。本书将会讲述与DNA有关的两个话题。其一，DNA本身也是一种对数据进行编码、数字化存储的方式，它的出现要比硬盘早得多。其二是一个与之相关的故事，测定DNA中的数据并确定其含义的浩大工程。

用DNA存储数据

基因组是生命体的数据库。它由DNA分子写成，其副本存储在人体的每一个细胞中（只有少数例外）。这种模式在自然界中普遍存在，即使是最简单的生命形式也不例外。基因组中编码的信息包含制造蛋白质的指令，而这些蛋白质组成的分子机器则运行着细胞内的化学系统。这就是为什么我认为DNA是可容错、冗余的存储形式。

你身体里的几乎每一个细胞中都有一个数据中心，数据中心里存储着这些基因数据库。我们称这个数据中心为细胞核，细胞核中又含有染色体。所有的人都是双倍体，你也不例外。你的每一个染色体都由两个副本构成，分别来自于你的父亲和你的母亲。除了普通的染色体之外，人类还有性染色体。女性有两条X染色体，男性有一条X和一条Y染色体。这些基因数据存储库的基本存储单元是DNA双链，两条链通过神奇的双螺旋结构缠绕在一起，如图15-1所示。

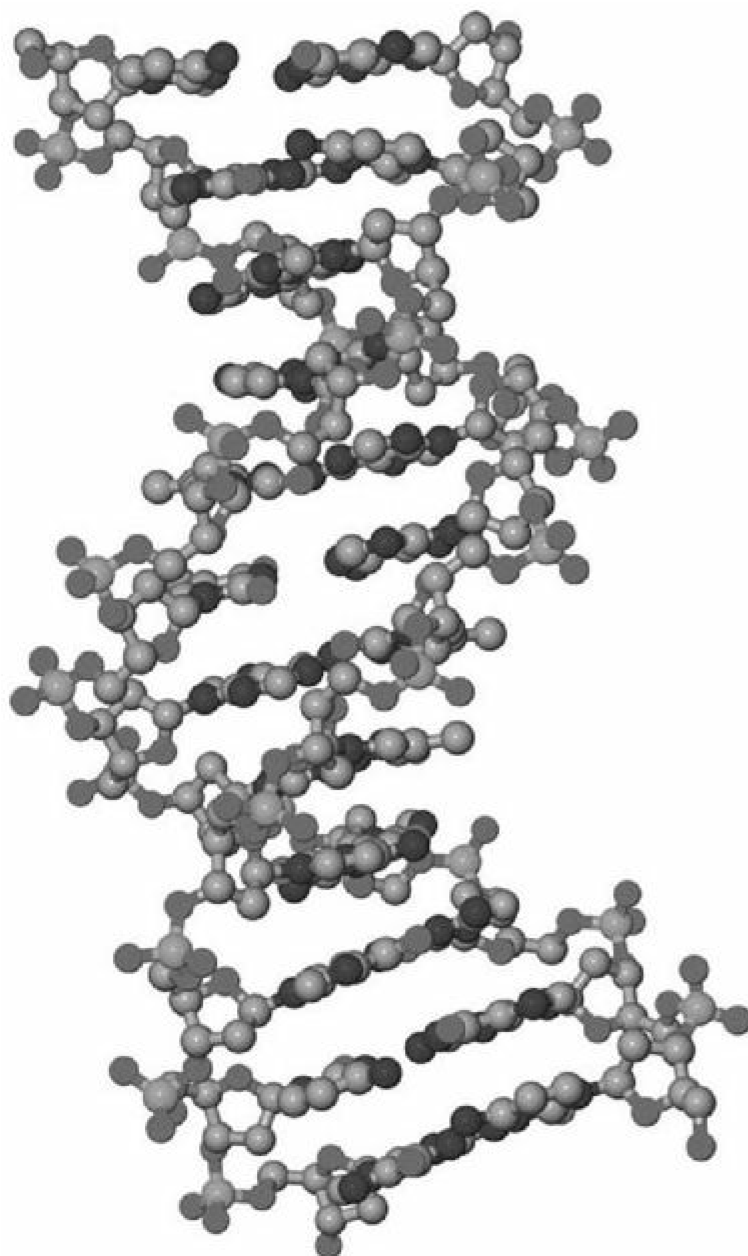


图　15-1：DNA中的一个小片段，由POV-Ray软件从PDB文件1BNA生成，doi：10.2210/pdb1bna/pdb（见彩图51）

DNA的每条链都是由一串碱基构成的。DNA中含有四种碱基——腺嘌呤、鸟嘌呤、胞嘧啶、胸腺嘧啶（或者分别缩写为A、G、C、T）。基因数据库正是编码存于这个四元体系中。在人体中，DNA含有

3G（3×10^9）个碱基，分布在两个稍有不同的副本中。

DNA生RNA,RNA生蛋白质

DNA到底是用来做什么的？人类基因组中的大部分似乎并没有直接的作用，尽管它们可能有一些潜在用途。基因组中的2%构成基因，基因的序列决定了蛋白质的构造。DNA是由四进制编码写成的大分子，同样，蛋白质是由二十进制编码写成的稍小的分子。组成蛋白质的序列编码确定了蛋白质的形状，而蛋白质的形状和化学结构又决定了其在细胞的分子机器中担当哪一个齿轮的角色。

和其他文件格式一样，读取和显示DNA序列中所编码的信息也需要特殊的读取装置。和其他种类的文件不一样的是，这个装置是在纳米尺度下在活细胞中工作的，它根据基因中碱基的排列，按照一定的顺序查找各个分子。RNA是DNA的短命表兄，它有一个类似四进制代码的系统。当一个基因被激活时，蛋白质分子机器系统会在RNA中生成该基因的副本。这些RNA副本被转移到蛋白质机器中翻译成蛋白质，每个RNA分子都能用作许多蛋白质分子的模板。

如此看来，DNA和蛋白质形成了一种伙伴关系：蛋白质提供机器装置，在基因中的碱基序列指导下制造新的蛋白质。虽然DNA被认为是“生命的蓝图”，但它也只能在细胞中分子（尤其是蛋白质）已经存在的情况下发挥作用。

通过控制负责激活基因的蛋白质的产生，细胞可以用产生蛋白质的方式对刺激物或者其他需要作出反应。通过这些机制，其他细胞可以通

过发送信号分子（比如激素）来控制某个细胞。许多药物也通过这些机制产生作用。

每一个基因都由其周围控制蛋白质产生的序列做了标记（见图15-2）。基因组中编码的一些蛋白质可以选择性地用于这些标记。一些蛋白质会促进基因的表达，而另外一些则会阻止该基因的表达。



图　15-2：图示为一个DNA片段，其中包含一个基因，以及基因周围用来和细胞调控机制相互作用来限制基因表达的区域。这里基因由一个启动子和一个增强子引导，这样基因就被做了标记，使得细胞知道何时应当表达该基因（见彩图52）

使用药物Hack你的DNA数据

让我们来幻想一下：O'Reilly要在“Hacks”系列图书中增加一本《DNA Hacks》——其中包含100种方式来修改、重构或者增强你的基因数据库。那么这些Hack方式的一半差不多都通过阻断或者激励某一类蛋白质的活动来进行，这类蛋白质被称作G蛋白偶联受体（G-protein coupled receptors,GPCR)。例如，胃酸分泌由体内信号作用于一个特殊的GPCR来激发。当信号分子触及GPCR时，将会发生一系列的相互作用，以至会“打开”或“关闭”你的DNA中的基因。例如用来治疗胃溃疡和胃痛的雷尼替丁，通过在信号和GPCR之间横插一脚来阻断信号，从而发挥作用。和在Email客户端中使用智能的hack来过滤垃圾邮件一样

（比如过滤包含“V1Agr4”的消息），雷尼替丁就是一个智能的hack，人们通过它来阻止传递“请分泌更多胃酸”信息的信号穿过细胞屏障。

还有一种形式的DNA元数据(mtadata)值得一提。尽管基因组通常情况下被看做一个只读数据库，但是DNA可以因为环境因素被化学修改，导致某些基因被抑制。这些化学变化并不改变基因自身的编码，而是通过提供附加的注释使得细胞机器不再容易转录该基因。研究这些化学修改的学科叫做表观遗传学。对于这些修改在身体组成中所扮演角色的研究现在仍旧处于初级阶段。不过目前已知的是，一些表观遗传学变化会关掉在某个特定细胞中不需要的基因（肌肉细胞需要的基因和肝脏细胞并不相同）。基因可以因为环境因素而被关掉——这似乎毫无疑问地会在幻想中的《DNA Hacks》一书的未来版本中出现，这确实让人激动不已。例如，生长素基因可以因为饥饿而被关掉。更有趣的是，这些变化可能会遗传给孩子，换句话说，我们的生活方式产生的变化可以影响到下一代的基因，这在以前被认为是天方夜谭。

癌症

癌症是一种由于细胞增生失控而产生的疾病。一个细胞的生长和位置是由它的基因控制的，也就是说，它只会以可控的方式分裂。除此之外，人类基因组中包含了很多肿瘤抑制基因，它们可以在细胞不受控增生的时候使细胞消亡。那么癌症是怎么发生的呢？答案就在数据库损坏中。化学致癌物（比如香烟中的）、辐射和病毒都能够使你的DNA产生变化。

当计算机硬盘碰到小损坏的时候，你可能注意不到有什么不同。不过当一定数量的损坏不断积累，程序就会开始崩溃，文件就打不开了。基因组同样如此。损坏足够多之后，基因的控制会产生紊乱，导致不正常表达的发生。如果你能触及正确的基因，那么细胞会开始向它自己发送生长和分裂的信号。其他的基因（比如肿瘤抑制基因）可以被完全关闭，一旦如此，你便在一步步走向癌症。

当然，硬盘在大多数情况下是可靠的。大多数硬盘中都有专门的系统来检测和重映射坏道。硬盘中也包含了检测磁盘损伤并通知用户的系统（比如SMART监控）。为了获得额外的可靠性，你可以通过RAID1阵列来将两个硬盘相互镜像。这样，两个硬盘的内容是相同的，如果一个硬盘出现错误，另一个硬盘便可以替代它并恢复数据。细胞自身也有检测和修复DNA损伤的机制。让我们来回忆一下，DNA分子以双螺旋的形式存在，一条链上的碱基与另一条链上的对应碱基互补配对。如果一条链发生损伤，它可以通过另一条链作为模板进行修复（类似硬盘的RAID1方式）。对于更大损伤的修复，还可以采用其他机制。

如果损伤不能修复，肿瘤凋亡基因就会启动，阻止细胞分裂并最终导致细胞程序性死亡。在理想的情况下，DNA受到损伤的细胞可以得到修复，类似硬盘的SMART机制。涉足过计算机集群（比方说5000个硬盘的规模）的人都会知道硬盘故障是多么的频发。想想人体内有100万亿个细胞，他们大概会奇怪为什么癌症的发病率会比较低。

复制

前文中已经说到，DNA是由缠绕在一起的两条链构成的。两条链是互补的：DNA的四种碱基都有对应的互补碱基（A和G互补，C和T互补），互补碱基出现在互补链的相同位置。沃森和克里克[1]在他们揭示DNA结构的论文中有一段著名的论述："我们注意到我们的碱基互补配对假说直接给出了一种可行的基因物质复制机制。"

每一个DNA链都可以用作另一个链的模板，用以复制DNA双螺旋。细胞可以分离两条链，将他们分别作为模板，用互补碱基制造新链。在细胞分裂为两个新细胞时，上述过程就会发生（见图15-3）。

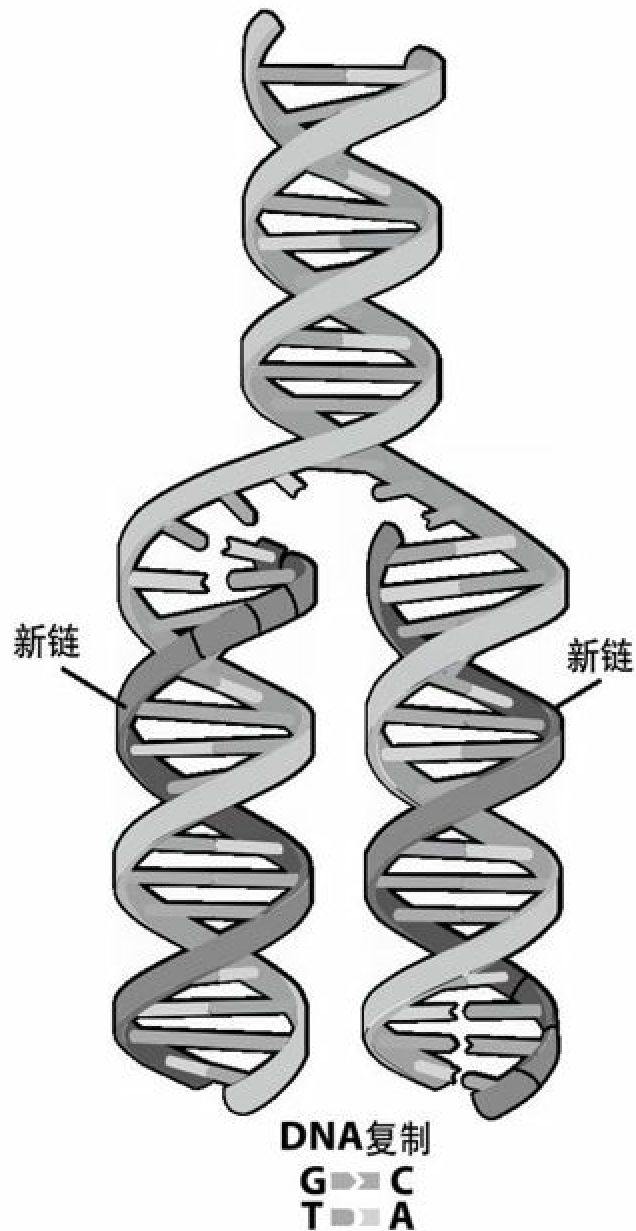


图　15-3　使用两个原始DNA链（白色）作为模板制造两个新链（黑色）的DNA复制过程，产生了两个新的双螺旋分子（照片来自http：//genome.gov/glossary.cfm)

破解基因编码

假设未来文明出土了一块21世纪的硬盘。即使未来人能够理解文件

系统（如前所述），文本文件在硬盘上最终还是以二进制形式储存的。如果没有将7bit数据转换为一个字母的7bit ASCII码表，信息就会无法解读。

1961年，克里克和布伦纳[2]开始了基因中类似的编码的逆向工程。ASCII将二进制转换为90个左右的字符，每个字符必需7个比特，即$2^7=128$（实际上还有一位用于奇偶校验）。在DNA中，四进制的碱基代码需要编码21个字母（20个氨基酸和1个终止信号），那么每个字符就需要3个碱基（$4^3=64$）（见表15-1）。

		T	C	A	G		
第一个碱基	T	Phe	Ser	Tyr	Cys	T	第一个碱基
						C	
		Leu		终止密码子	终止密码子	A	
					终止密码子	G	
	C	Leu	Pro	His	Arg	T	
						C	
				Gln		A	
						G	
	A	Ile	Thr	Asn	Ser	T	
						C	
				Lys	Arg	A	
		Met				G	
	G	Val	Ala	Asp	Gly	T	
						C	
				Glu		A	
						G	

举例来说，三个碱基ACT编码了苏氨酸。当翻译机器在基因中遇到ACT三联密码子时，就会向制造中的蛋白质分子插入一个苏氨酸。我们可以注意到，密码子的数量（64）比信号（20个氨基酸和一个终止信

号）要多。这意味着三联密码子有内建的冗余机制——大多数的氨基酸由不止一个密码子编码。

实际上，对大多数密码子来说，最后一个碱基对编码没有影响或者影响很小。这使得碱基突变（碱基的变化，比如A变为T）对蛋白质造成影响的可能性变得尽可能低，也就保护了蛋白质结构免受可能的碱基损伤影响。

人类基因组在有意外错误并入的情况下已经成长了几十亿年。因此，基因组中充满了没有作用功能的元素以及那些作用功能不明确的元素。虽然“垃圾DNA”这个说法现在已经不再流行了，但是不可否认的是，DNA中大量信息的存在仅仅是因为将它们移出基因组的选择性压力太小了或者根本不存在。

有些元素可以自行在基因组中复制、粘贴。例如，Alu元素，一个大约有300个碱基的碎片，其中包含的信息可以它自己复制为RNA（一种寿命短一些的类DNA分子），然后在其他位置重新复制进入基因组。这样的结果就好像让库布里克(Sanley Kubrick)的《The Shining》中的人物杰克和你一起写本书，书中写着“只工作，不玩耍，聪明杰克也变傻”[3]。Alu元素占到了人类基因组的10%。简单生命体基因组的大小限制了这些元素的增殖，人类基因组中这些元素的存在也就使得人类基因组更为深奥难懂了。

DNA作为数字化存储

进化的力量塑造了基因组。但在达尔文提出进化论的时候，他知道

自己碰到了一个问题。常识告诉我们，高个子和矮个子生出的孩子身高会一般，那么“高”和“矮”的遗传信息会不可逆地混合在一起。有性繁殖会让一个物种达到平均水平，但事实上，正如和达尔文同一时代的孟德尔[4]所发现的，基因并没有不可逆地混合在一起。

原因在于DNA的数字化本质。中等个头的子代会有来自父亲和母亲各一套基因拷贝。基因并没有混合在一起，举例来说，序列中没有任何一点有半A半T的状态。所以信息并没有丢失，并且可以原样传递给孙代。复杂的生命得以进化并存在，这就是由DNA所实现的数字化储存。

进化作为算法

我们知道创造细胞中DNA的基本算法是：进化。DNA序列因变异而变化。那么进化是如何塑造DNA？令人吃惊的是，进化（以两代间DNA序列变化定义）的大部分是选择中性的，即进化对生命体没有任何影响。为什么是这样？很明显，在小种群中这种变化的发生有机会以后在种群中占支配地位。对大一些的种群来说，这种变化占支配地位的可能性较小，但是这种变化的数量会相应增加——所以最终中性进化仍然占优势。

这对解释个体间与物种间的差异很重要。中性进化是两个基因组之间任意不同的空模型(nll model)。这意味着，为了证明两个物种的相同基因中存在的差异是有意义的，科学家们一直想证明中性进化的“算法”无法生成两个序列。而这需要估算变化的比率，并确定其是否会快于或慢于中性理论所预计的速度。如果产生了多于预计的变化，则可能

是由于这些变化已经为组织提供了功能上的改进，并且这些变化已在种群中迅速蔓延。如果变化比所预计的要慢，那么基因可能足够重要，使得绝大部分变化不再是中性的，然后被进化所拒绝。

人类参考基因组[5]目前被看做表示不同人混合的线性字符串，物种内的巨量差异意味着人类参考基因组可以更适当的被看做包含了种群内不同的图(Gaph)。一个个体的基因组可以被表示为参考图的一个遍历。

[1]即詹姆斯·杜威·沃森(Jmes D.Watson)与佛朗西斯·克里克(Fancis Crick)，此二人因发现DNA分子双螺旋结构于1962年获得诺贝尔生理学或医学奖。

[2]悉尼·布伦纳(Sdney Brenner)，南非生物学家，2002年诺贝尔生理学或医学奖获得者。

[3]“All work and no play makes Jack a dull boy.”是《The Shining》中的原文。

[4]格里哥·孟德尔(Gegor Mendel)，奥地利遗传学家，遗传学的奠基人。他通过豌豆杂交实验，提出了遗传因子（现称基因）及显性性状、隐性性状等重要概念，并阐明其遗传规律，后人称之为孟德尔定律。

[5]Reference Human Genome，即人类基因组计划所测序的基因组。

DNA作为数据源

用编程语言来说，DNA就是一个字符串：

char（3*10^6）human_genome；

人类的完整基因信息由30亿字符组成，即使在最没效率的自制编程语言中也可以很轻松地处理。但是确定这30亿个碱基的确切顺序需要化学、生物信息学和实验室过程的通力合作，当然还需要很多旋转的磁盘。

人类基因组计划第一次将目标设定为对所有这些字符进行测序。全世界很多大型、高产的研究所都将学术竞争放到一边，着手这项可能持续13年花费数十亿美元的任务。他们的目标是制作出一个健壮、精确的人类基因组图谱，并向所有人免费发布。英国、美国和日本的科学家联合组织已于2001年2月在科学文献上成功发表了第一个人类基因组草图。如果不算附加注解和相关数据，基因组本身有10Gb的数据量，这在没有iPod或者U盘的时代是相当大的。然而，随着数据在世界各地的分发拷贝，对储存空间的要求以指数形式增长，因此整体的数据要大得多。科学家着手分析数据，将其用作基因标记与疾病指标的来源，并将其与其他来自老鼠、酵母菌、病菌的基因组对比。这10G数据已经构成了现代生物研究的基础。

现在将时间快进到2008年。人类基因组已经得到了很好的注释(http：//ensembl.org)，包括老鼠、黑猩猩、鸭嘴兽在内的其他四十多个

物种的基因组也被测序了。在威康信托基因园区[1]中摩根楼一角的会议室中Sanger研究所的IT专家、信息学家和科学家正在一起开会讨论DNA测序的下个时代。

若干互相竞争的测序技术平台已经给出了数字，并都期待着去探索基因信息的非富余本质。与已经被最初的人类基因测序测出的碱基对里的长读序列不同，这些短读序列将只有30～50个碱基，而前者则数以千计。但这30～50个碱基足以在特定的染色体上的特定位置进行读取，而这在多达30亿字符中有很高的确定性。数以百万计的碱基迅速使得在毛细血管状的基因序列中计算单个Kb变得逊色不堪，因为短读序列的数据要求的细节已经十分明了。白板上填满了手写的图解过程，那些过程正是运用了荧光碱基的图像，用来确定这些DNA短片段的正确序列。

人们已经讨论过了要支持研究所的科研目标所需的仪器通量指标及仪器数量。白板上的一个数字与其他数字相比，更加引起了人们的注意。在以后六个月中，Sanger研究所将会测序出50Tb的最新数据，是每星期50Tb。

会议上一片寂静。

即使我们有能力支持大数量序列的人类基因测序仪器（100+），同时也拥有可供分析和注解数据的专用的现成硬件，但是整个生物学界也从未处理过具有如此原始信息的数据。

新的测序平台将会大大地超越Sanger研究所现有的大规模数据需求。也就是说，数据需求已经在Sanger研究所15年的历史中飞速呈指数

增长，最终达到了由17000个在线硬盘驱动的PB级储存系统。在未来12～18个月内，新的测序技术会产生相当于这15年总和的数据。

果不其然，测序仪器在基因园区里掀起了新一波的浪潮，6个月后，那场会议的寂静被工业史上最大规模信息设施打破了，有服务器农场、存储阵列、软件、数据库和信息管理工具。

量的飞跃

新技术在基因测序中产生了量的飞跃。上一代的仪器需要大量设备来产生1Mb的可用序列。测序一个单一的基因组就要花费好多年并且费用昂贵。

测序一个基因组在比对不同物种的基因组时是很有用的。DNA序列中有生物学意义的部分在进化历史中保持不变，而且找出这些不变的区域可以帮助生物学家识别新基因，新基因则可能在疾病中扮演某种角色。

由于单个Kb的测序很少是完全精确的，在统计学效能上，对同一碱基进行多个相互独立的测序是有切实好处的，不过这样就需要大规模的部署仪器。不过新一代的短读测序仪器每次运行都会读取同一碱基几百万次，可以为结果基因组提供额外的解析度，这对比较基因组很重要。

有了这个额外的解析度，短读测序方法第一次使得构建大量个人基因组成为可能，从而提供了个人基因组数据库的洞察力。将这些基因组进行比较，可以帮助识别出为什么有些人比别人有更大的风险患高血压或者乳腺癌。基因序列中很少一些碱基的差异就能改变这些易患病基因

体质；从一个单独的碱基(sngle nucleotide polymorphisms,SNP)到重复单元集合(cllections of repeating units,CNV)，30亿碱基中的这些小变化正是很多疾病状态的关键。通过比对短读技术获取的几百万份拷贝DNA字符串中某些特定位置的碱基，就可以肯定两个体碱基差异是测序错误引起还是真的有微小的核酸变化。通过病例和其他的研究进行额外的注解就可以知道这些变化是不是和疾病有关。

全是碱基

这种基因测序平台的技术实现值得讨论一下。从本质上来说它是线性组装流水线，其中每个DNA都是原材料，计量后在测序仪器中准备参与光化学反应。将DNA剪切为数以百万计的短片段后，再将其复制几百万次，然后附着在一种特殊制备的玻璃片上。带有特殊标记的碱基会堆积在这些制备好的短片短上，并配对在互补碱基上：A对T,C对G。用激光照射时这些碱基会发出不同波长的荧光，然后就可以拍到含有几千个点的图像，每一个点都是一簇发光的DNA碱基（见图15-4）。此时对测序平台的要求就以乘法方式增长了。每张图像都只是百万像素的解析度，采用短读测序时每个碱基要拍四张图像（每个位置的每个碱基一张）。这也仅仅是148张图像。

不过，每张图像都只只能覆盖DNA簇的一部分。DNA会分成330个单独的块进行拍照（48840张图像），然后是8条线（390720张图像）。通常情况下，DNA链是由正向和逆向双向测序（781440张图像）。同时还有测序运行是由激光强度和射流测量产生的元数据。

一台仪器每天可以总计可以产生1Tb数据。

这些数据传给图像分析软件，软件将基于荧光强度通过校准每一系列的图像确定碱基顺序。这样又生成的文件含有各个字符串，质量评分和荧光强度详细信息，同时附带了簇位置和其他相关的元数据。实际上，为了方便下游的分析，这些信息是写成两种文件格式的：fastq和SRF，一种被全世界同事采用的数据格式。

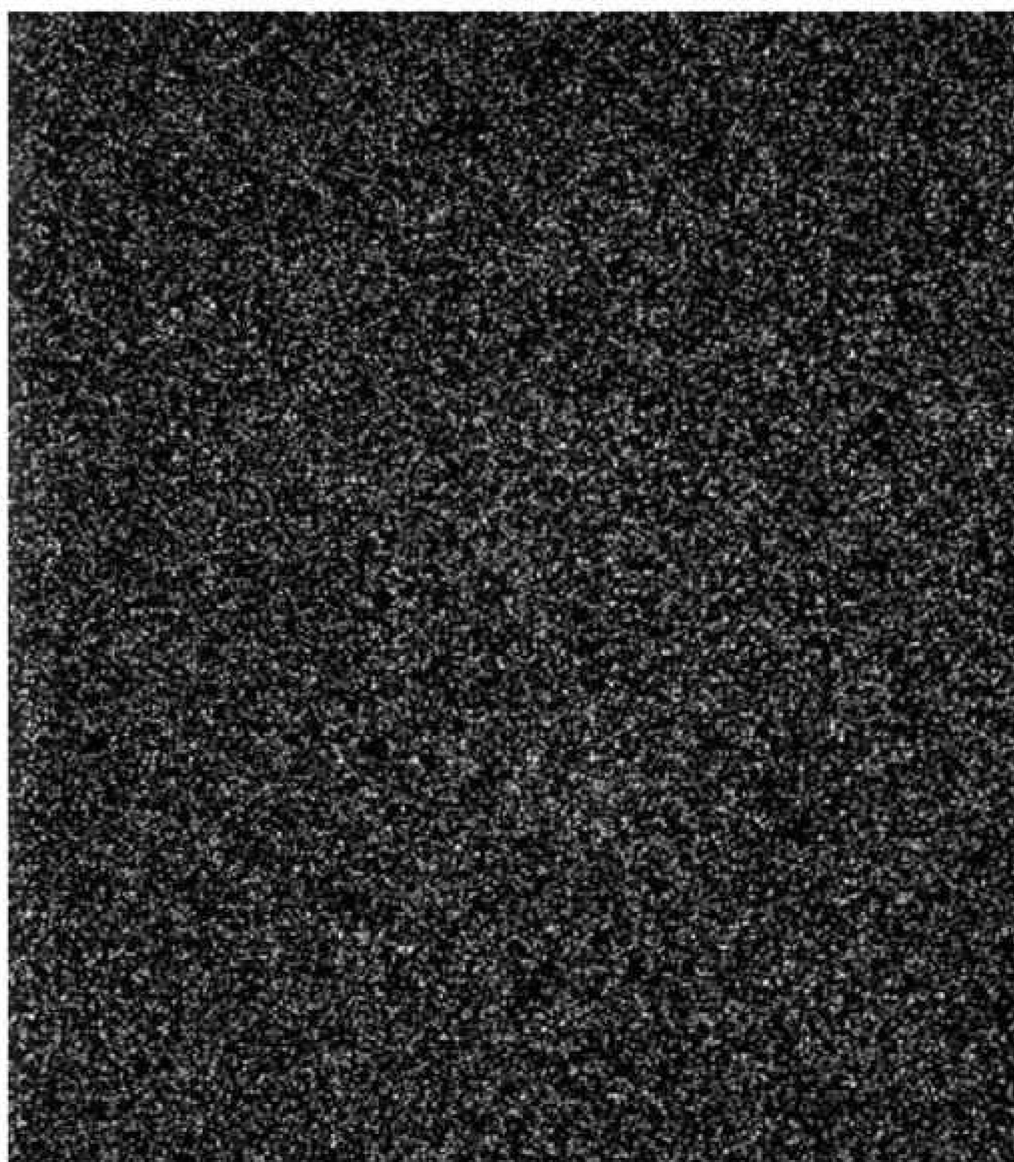


图　15-4：Illumina[2]GA2测序图像，其中每一个点都是由数千个DNA分子组成的簇，在激光下发出荧光

图像分辨率与读取长度的简单增加对数据处理和分析有着巨大的影响。Sanger研究所新测序平台的最初版本可以进行每次36个碱基的短读。100个碱基对的读取长度（比如Sanger研究所研发实验室里运行着的原型机）就会导致图像数目变成三倍（2334320张图像）。在拥有40台仪器的一个平台中（你读到本文的时候可能已经不止这么多了），每个平台的每台仪器总共会产生9300万张图像，每次运行大约持续一星期，总计每周会产生75Tb的数据。

数据是怎么处理的？

[1]Wellcome Trust Genome Campus，位于剑桥南部的Hinxton，是Sanger研究所和欧洲生物信息学研究所的所在地。

[2]Illumina是一家提供DNA测序仪器解决方案的公司。

搏击数据洪流

大规模的部署测序仪器对很多方面是不可或缺的，比如基因数据集建设，支持诸如千人基因组计划[1]和国际癌症基因组协会[2]这样的大规模、全基因组关联研究，这对今后20年的生物研究有着巨大的价值。世界各地的大型基因中心令人钦佩地接受了这个挑战。拿Sanger研究所来说，有超过35个Illumina GA2基因组分析器，运行在位于英国剑桥以南10英里处Hinxton基因园区的高通量设施中。

Sanger研究所的测序平台

测序平台是Sanger研究所的核心服务，对所有Sanger研究所或其他合作机构进行的基因组研究开放。由于测序对设备的要求极高，同时，Sanger研究所也开发了一系列操作工具与流程来满足这种高要求。

项目管理

我们开发了名为Sequencescape的一些列项目管理工具集，可以帮助研究者计划自己的实验，并帮助测序设备管理者计划容量和吞吐量。Sequencescape最初是被内部小组开发维护的。它用Rails语言编写，运行在标准的刀片服务器上，采用MySOL，通过内部网络(htp：//www.sanger.ac.uk/）供用户访问。

当一个新项目需要测序时（可以是单一样品的单次运行，也可以是1G碱基位上万样品的数千次运行），就在Sequencescape上登记，附带提供一些相关的元数据，例如科学理由、预算信息和联系方式。

每一个待测序样本都会以这种方式登记，测序运行的请求提交到设备。这个请求组成了测序平台工作的基本单位，物理DNA样本则被送交实验室小组来进行准备和测序。

准备测序样本是一个复杂的手动过程，其中包括一个不断提纯的实验室流程。物理DNA样本在手动仪器、自动机器和计算机控制组件间，甚至不同实验室之间，从一个试管中移到另一个中。记录一个样品的“生命故事”是很关键的，在评析测序结果、实验室吞吐量以及确定实验流程组件的错误调校或故障时，这些元数据可以导入进来。样本中的DNA如果稍微处理不慎，仪器就不能测序：这是一个费用昂贵并且耗时冗长的问题。通过评估和分析元数据，可以在失败中看到共同性，找出通讯协议偏差、误运转的器械或者错误调校。在不懈的监测和评析中，实验室进程得以继续提供高质量DNA样本以供测序。

生物实验过程，用俏皮的生物术语来说，是一个持续进化的过程。随着科学期刊或者会议中新方法的讨论发布，优化、更新、采用全新方法，或者是彻底推倒重做都是很常见的。新仪器和机器自动化引进使用。在实验室工作流这种高灵活度领域中，保持健壮的原始数据获取是极其困难的。它也很快在高吞吐量环境下变得不可实现，在这个环境中，即使很小低效或者停滞也会导致长久的事物堆积。

毋庸置疑的是，基因测序中病人个体的基因材料也许有限、供应不足，或者保存期限短。制备好的待测序DNA也不能永久保存，而且对于不能再次获得的DNA，实验过程或测序的延迟将会带来灾难性的后果。

为了平衡对有效原始数据集的需求和时间紧迫且高度动态的工作流之间的矛盾，现代的数据获取方法必不可少：我们不想看到软件成为工作流程中的限制步骤。

灵活的数据获取

传统的数据库重构，都是对数据模式(dta schema)进行增、删或者重命名，然而Sanger研究所的高通量生产线使用的数据建模工具集却允许在运行时改变数据描述。当一个实验室协议升级，包括一系列新仪器或者快速方法时，与任务相关的现有原始信息可以在运行时重构。如此帮助实验室更有效率地运作，减轻开发者时间负担，还使得数据集可以与当前实验室流程保持一致。

对大多数部分来说，生产系统中运行的数据库仍然是传统型的，存在于诸如项目、样品和工作流、任务这样关键协议组件的真实世界对象中。不过，元数据也被定为主要对象，表示为描述符(Dscriptors)、值和族(Fmily)对象类。一个族可以有多个描述符，描述符提供数据字段的定义，包括名称、类型、UI元素和是否为必须字段，等等。每一个工作流都可以有效继承一个族的字段。

这就允许我们快速定义和修订新类型的项目、样本和工作流任务。

所有的数据项页面的UI都是使用这些信息动态生成的，用以监测测序平台和实验工作流通量的多个报告工具也是一样动态生成的。族有版本控制，这就使得老项目数据可以持续访问，同时也有迁移和校对工具将旧版本对象升级为新版本。

从计算角度来看，这个过程比我们预想的要快。借助数据表中足够的索引和数据范式化，即使高通量系统高负荷运行时Web应用程序也能迅速反应。由于某些属性是牢牢固定在工作流中的（比如，每个物品都有名称、唯一标识符，以及创建和修改日期），他们可以很容易地重构回主要数据库字段以提供更快速的查询。

这项对于实验室工作流数据模型进行精简的尝试已经取得了很大的成功。优势越来越明显，因为灵活度的优化可以让相同的构架应用于研究所内其他实验室工作流中（首先用在了基因型鉴定中），而只需要很少的额外开发。Sequencescape项目管理和实验室信息管理工具是开源的，可以从http：//www.sanger.ac.uk下载。

仪器和数据管理

在Sanger，整个短读测序流程当做了一系列托管服务来运行，和Web服务定义的界面有通用性。项目管理和实验室信息管理最终让位于仪器管理工具，此工具可以监管、控制整个仪器平台，并尽可能地实现自动运转。

每台测序仪器都连接着一台PC，电脑用来控制仪器并给仪器产生图像数据提供临时存储空间。遗憾的是，尤其是对长一点的测序来说，本地机器的磁盘空间会迅速变得不够存储整个运行的数据，而且桌面PC的计算能力不足以进行数据的图像、序列与质量分析。

因此，数据通过一个10G b线路从仪器上连接的P C传输到更大的存储阵列中：由Lustre[3]管理的400Tb大的EVA存储系统中。这便是一个

1000节点的网络集群，此集群承担了原始图像的主要分析和图像校准任务。管理这个规模存储系统问题很多，需要管理员不断干预，供应商不断扩展还有内部数据管理者的监管。这个阵列处理了差不多4个星期的数据；随着分析和最终的碱基识别，仪器中的原始数据相关的仪器运行细节（激光强度、荧光数据等）在删除之前都按需进行了校对并存档。

这种规模的原始测序数据很大，处理费用很高；不过，图像校准、序列上个各个碱基确定之后，每次运行产生大约30Gb序列和质量数据。如果原始数据没有备份（从磁带上恢复数据需3个月），每一个测序运行的原始图像压缩成低质量的JPEG文件，保存在数据库中。尽管不适合分析，如果任何一次运行都需要人工判读来确定是图像问题或者人工制品（油，不太好的DNA集群，甚至指纹都不稀奇），这个数据仍是有用的。

一旦测序数据可用，它会以两种格式被存储到一个高性能的Oracle数据库里。虽然生产系统充分利用了数据库，但是生物信息学工具通常会在文件系统上用普通文件工作。我们确实需要迎合所有的需求，这个存档里大量序列信息也同样通过一个Fuse[4]用户空间文件系统呈现给Sangery研究所内的计算农场。

这项措施出人意料的好。序列数据紧接着通过一系列的质量控制措施传送出去，然后在序列分析集群上运行，检查低测序产出、高层次的未知碱基，或者低复杂序列，而所有的这些都是测序错误的标志。质量控制(Q)结果又被传回到Sequencescape，研究者和合作者会收到有关新

数据及其在Fuse资料系统位置的提醒。在测序结果并不具有足够质量，或者如果实验室存在问题的情况下，Sequencescape会自动排队样本进行再处理。

对Sanger研究所的许多项目来说，得到许可之后，数据会对所有人开放：从最初的项目和样本元数据，到实验室的原始信息，以及从每个样本中获得上百万测序数据。这些材料任何感兴趣的人都可以免费下载，包括最新的基因数据草图(htp：//www.ensembl.org/index.html)，各个测序追踪结果(htp：//trace.ensembl.org/），随后几个月，还会有更多的项目的短读序列（请关注http：//www.ebi.ac.uk)。这就为生物医学研究者提供了优秀的资源，同时，也继续了为人类基因组计划提供免费开放数据的光荣传统。

[1]即1000 Genomes Project，此计划将测定世界各地至少1000个人类个体DNA序列，参见http：//1000genomes.org。

[2]即International Cancer Genome Consortium，参见http：//1000genomes.org。

[3]Lustre是一种分布式文件系统，参见http：//lustre.org/。

[4]http：//fuse.sourceforge.net/

DNA的未来

这些新的测序技术和数据作为使用它们的工程的一部分正在为现代生物学的下一个篇章建造基石。运用此数据的管理工具、校对工具和分析工具将会继续进化，因而花费稍微长一些的时间关注DNA的未来，是十分有价值的。

如何成为基因黑客

与其他数据密集型领域相比，基因具有更加伟大的提供开放在线数据仓库的历史——从种类多样的基因浏览和注解工具（例如Enesembl和UCSC)，一直到和基因有关的疾病(HpMap,SNPedia)细节，例如23andMe和Navigenics的私有化的基因服务。这里有足够丰富的资源以供任何人日后成为一个基因黑客。

下下一代

目前，这种公司仅仅提供对基因感兴趣的几点高层抽象。但是创新一如既往地伴随下一代测序仪器和基因分析器的发展而前进着。像Pacific Biometrics和Oxford Nanopore这样的公司正在致力于可以使目前的兆碱基(mgabase)级的读取转换成千兆级(ggabase)，甚至更大。随着吞吐量的提高，相关费用会下降，成本为$5000，甚至是$1000的基因测序，对治疗学基因测序来说，这些已经变得很经济有效，并离我们越来越近。

大数据时代

以上的尝试若有所进展，我们可以肯定一件事情：数据需求正在急剧上升。大数据时代对于基因来说已经到来，这对于现代生物研究也是一样。数据将继续提供极其巨大的约束分析和研究，而且，很明显，越来越多的研究和医院队伍进行内部基因测序，大基因中心的角色将会有所改变。为成千上万的基因提供一个灵活的计算平台，有效率的序列搜寻、排列和汇编工具，加上安全的环境，将会变得至关重要。这也会为它们带来技术实现上的问题，比如隐私以及效率，所有这些都将会在当今生物医学的舞台上成为焦点话题。

尽管这些阻碍也许会带来一些地平线上的未知云团，但DNA的未来必定光明无限。

致谢

作者在此感谢Wellcome Trust对生物医学研究中开放接入和开放数据的不懈支持。

第16章　美化真实世界中的数据

Jean-Claude Bradley、Rajarshi Guha、Andrew Lang、Pierre Lindenbaum、Cameron Neylon、AntonyWilliams和EgonWillighagen

关于真实数据的问题

要在真实世界中搜集“数据之美”，并将其展示给感兴趣的公众，至少有两个问题需要解决。第一个问题：世界本身是嘈杂的。在大多数情况下，重复搜集同一块数据两次会得到不同的结果。这是因为搜集过程不可能完全无误差。温度、压力、湿度、动力源、水或试剂的质量、称量精度的波动，还有人类的误差都会使“正确”结果变得模糊。实验测量的艺术在于如何设计数据搜集过程，使随机误差与操作错误对结果的影响最小化。在最好的情况下，这涉及如何精确优化实验设计，检测误差的大小及来源。在最差的情况下，人们就只有不断重复实验，直到得到他们满意的结果为止。

对于处理误差导致的不确定性，传统实验的做法是进行重复实验并将结果付诸统计分析。重复的例子可以在大多数科学期刊的大多数文章中找到，很多图表中有“典型结果如上所示”。“典型结果”一般意味着“我们获得的最好数据集”。尽管理论上详细的统计分析是一种更严密的方法，但是也可能存在争议并让人误解。医学期刊的评论页中常常对从分析中删除不相关数据的适当方法产生争议。之所以对“典型”结果产生怀疑，对统计方法怀有争论，是由于无法接触原始数据。如果底层数据可得，人们自己便可很简单地进行分析和检查。虽然争议似乎不会因此减少，但是至少人们可以更加知情。

这个问题的第二部分是，一直到最近，印刷期刊的版面限制了可发

表的数据总量，如此，便使得支撑论文论点的完整数据及数据分析的发布变得困难或者根本不可能。然而，在出版开始逐渐转向线上的当今世界，这个理由变得牵强了。起码在数据量处于KB级至GB级的研究中，将论点所依赖的完整数据集发布出来是可能的。因此一直有很高的呼声来要求发布完整的数据。不过，这样就产生了如何呈现数据的问题；数据可能不一致，也可能包含错误，但是起码可以显示出所得结论的总体描述。简单地说，问题就是：如何清晰地展示出隐藏在数据表面下的美丽，同时又可以不刻意避免或者隐藏数据表面的瑕疵。

我们认为要成功调和这些相互矛盾的要求，关键在于透明。以尽可能全面的方式提供原始数据，并处理和筛选数据的完整描述，意味着任何用户都可以深度挖掘到他或者她需要的细节。原始数据通常很难或者不可能自然而然地以一种机器可读、可处理的形式发布，所以筛选和提取过程同时也包含了对分类和简化的决策，从而提供清楚明了的数据以作其他用途。这里描述了我们在“美化”一系列开放来源[1]数据时所采取的方法，以一种开放的形式筛选和发布数据，允许任何人将其收为己用。结果表明，以上描述的方法已经使得多个研究人员能够配备一系列可视化和分析的工具，创建一个协作网络来有效地分析结果、提议后续实验，并将结果发布给更广大的读者群，而这皆是传统研究通信所不允许的。

[1] 开放来源即crowdsource，指的是一种任何人都可以提交数据和参与各项任务，也可译作“众包”。

提供可以追溯到记录本的原始数据

作为Jean-Claude Bradley教授所指导的一个药物发现研究大计划(Badley 2007）的一部分，我们希望预测各种各样的化合物在非水溶剂（比如乙醇、甲醇等）中的溶解度。Bradley研究组用来合成潜在抗疟疾目标化合物(Badley等2008）所用的Ugi反应主要成分有醛类、羧酸、异腈和基本氨基酸，这些物质的溶解度是最让人感兴趣的。化合物的溶解度是化合物可在某个特定溶剂中溶解的数量。构建并验证一个可以预测溶解度的模型需要大量溶解度数据。出人意料的是，世界上并没有唾手可得的非水溶剂溶剂度数据。因此我们选择开放数据来源，将数据测量开放给想要参与的任何人(http：//onschallenge.wikispaces.com/）。不过这样就产生了一系列问题。由于任何人都可以提交测量值，我们没有直接的方法来检查测量值的质量。

因此创建数据集的第一阶段需要创建如何测量每一个数据的详细记录。不同贡献数据的测量技术、精度和准度都不一样，但是所有的背景资料都以人类可读形式提供。这种将完整研究记录在实验完成时就发布出来的“彻底分享”方式称为开放记录本科学(http：//en.wikipedia.org/wiki/Open_Notebook_Science)。这种方式虽然在专业研究者中并不普遍，但是它很符合我们公开一个完整和透明的数据集的愿望。我们利用在Wikispaces托管的wiki(http：//onschallenge.wikispaces.com)来保存实验记录，利用诸如

Gdoc和Flickr(http：//flickr.com)等其他服务来保存数据（见图16-1）。

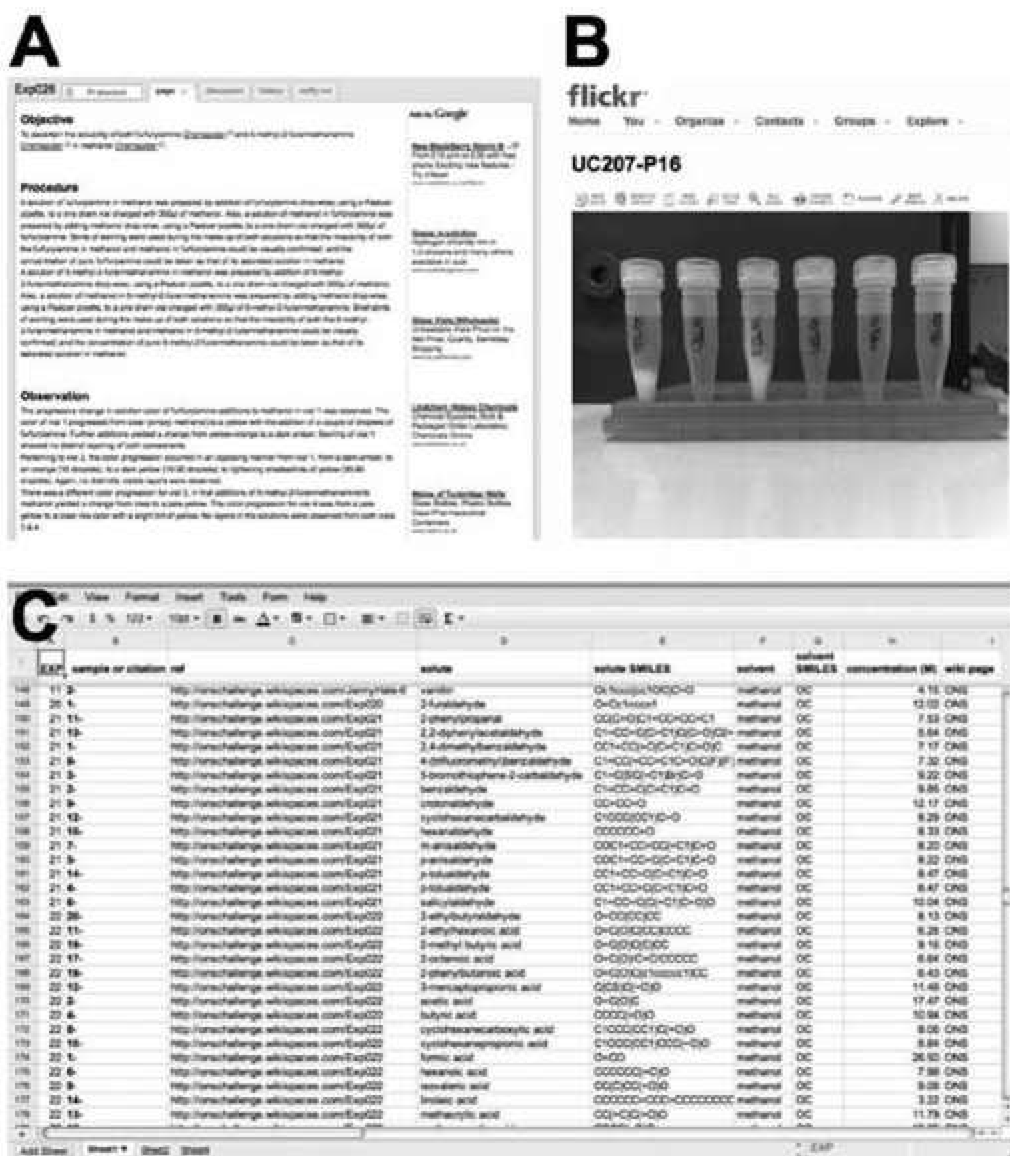


图　16-1：使用免费通用服务来托管实验工作记录及处理过的数据。A）单个实验测量时期的一部分；B）Flickr上实验中所拍照片；C）托管在Gdoc上主要数据仓库的一部分（见彩图53）

从实验描述中提取出数据值的数据库储存在Gdoc的电子表格中，用来生成项目的主要汇总数据。各条测量值一个不漏地发布在此，并附有一条指向原始数据的链接。无论是机读还是人们来阅读数据，这条链接

都是很重要的，因为它不仅提供了测量值的出处（即谁给出的断言），还提供了此断言证据的记录。人们可以点击链接来查看测量是如何进行的，机器则可以在必要时下载或者抓取这条记录。

在某种意义上，电子表格是从实验室记录本转换为可以评论和过滤数据的第一步。从技术角度来说，选择Gdoc可能是一种偏好，不过这是基于以下几点要求的：允许我们以原始格式发布和共享数据；有一个实验科学家熟悉的界面并不占用他们额外的工作；由稳定的大公司维护的免费托管服务，可以让世界上的任何人以最小的代价复制这个信息处理模型；最后，可以用强大并灵活的API访问数据。很少有其他的方式可以让普通科学家用熟悉的方式使用、添加、下载原始数据，并且也为其提供强大的底层数据程序访问。

验证开放来源数据

由于数据是由不同的研究人员采用不同的方法搜集的，因此很可能出现矛盾的值。这可能会导致出现异常值或者直接出现相差很大的结果。习惯上，如果没有额外的信息，研究人员只能将每个测量值给以相同的权重或者应用统计方法来剔除异常值。然而，由于我们采用了需要完整记录每个值测量过程的开放记录本方式，每个测量都可以通过记录的信息上下文来进行评估。在若干情况下，这样可以允许熟悉此方法的科学家在测量条件不适当或者忽略一个重要参数的事实基础上，剔除有问题的数据点。

就溶解度来说，事实证明：混合时间和蒸发条件都是重要因素。对硝基苯甲醛在甲醇中溶解度的确定就是一个很好的例子。在五个测量值中，有三个明显低于另外两个（见图16-2；http：//oru.edu/cccda/sl/solubility/ugidata.php?solute=4-nitrobenzaldehyde＆solvent=methanol)。测量方法基于对硝基苯甲醛的甲醇饱和溶液配制，然后蒸发掉甲醇，称量残留物重量。其中的关键就是完全饱和溶液的配制，配制过程通常是在搅拌条件下添加溶质，直到试管中有不再溶解的可见固体。通过检查这些测量实验的详细情况可以发现，测量值较低的三个实验都只是稍作搅拌，而测量值较高的两个实验则搅拌了好几个小时，这说明实验需要长时间搅拌(http：//usefulchem.blogspot.com/2008/12/mechanical-turk-does-solubility-

on.html)。

原始实验记录的可获得性使得所有研究人员都可以发现存在疑问的测量值，也可以从“失败”的实验经历中受益。这就是科学研究的本质。在记录实验细节和高效之间需要达到一个平衡。初始实验的目的通常是帮助研究人员找出需要注意的影响因素。遗憾的是，这些信息通常不会在研究社区内分享。

对于这些存在疑问的结果，通常我们不是直接剔除它们，而是在这些测量值上加上“请勿使用”的标记，并给出原因。这使得其他研究人员可以点击进入原始的实验室记录本页面并自己检查原始数据。错误可能发生在任何地方，甚至包括验证结果的过程中。“错误”的数据在某些用途上也可能是有价值的。完全透明可以让数据的用户来决定在分析中包含什么数据。这也同样会减少（但是不会消除）隐藏错误的风险。这里有一个这种标记的例子，即香草醛在甲醇中溶解度的测量报告(htp：//usefulchem.blogspot.com/2008/11/what-is-solubility-of-vanillin-in.html)。

显然，在我们标记为可疑或者不可信的数值和没有标记的数值之间有一片灰色地带，尽管未被标记的数值在有些问题或者领域还存在争议。一天下来，它们可能会得出科学的判断，同时，也会产生大量分歧。每个案例的实验原始数据都可以被获取与查验，而且电子表格的历史版本同样如此。我们需要保持以下两者的平衡：提供一个有用的数据和每个决断与错误的呈现程度。然而，这个任务是很具有挑战性的。

在线发布数据

我们的目标是让所有的实验记录与处理后的数据可以在线得到。如何在Web中以一种有用的格式发布数据产生了一系列的问题，其中包括标准标识符、可视化工具和数据集成方式的选择。

化学实体的唯一标识符

要让我们的数据有用，使用公认的标准来描述化学实体就很重要。如果不这么做，数据集的集成就会变得很困难或者不可能。在化学中，有些人认为使用CAS注册号[1](htp：//en.wikipedia.org/wiki/CAS_registry_number)标识化学实体很理想。不过，CAS号本质上是私有的，而且不能转化为化学结构，只能用作查询，而且依赖与一个外部的组织来进行编号。我们更倾向于本质上开放的标识符，可以自由地进行交换，并可以同化学连接表[2]双向转换。

IUPAC[3]国际化学标识符(IChI，发音为“INchee”）提供了一个非私有的标准与算法以及相应开源支持软件(htp：//en.wikipedia.org/wiki/Inchi)用于标识字符串的生成及标识符到化学结构逆转换（参见http：//www.qsarworld.com/INCHI1.php的最近讨论）。InChI作为一个标准正在获得软件厂商、出版商和开发者的鼎力支持。InChI算法——即对同一结构生成多个不同InChI的可能性——已经通过标准InChI(Standard InChI)的开发得到了解决。对于某些用途，

InChIKey(InChI的一种hash码）已经很实用，可是InChIKey不能转换成结构，而且只能在搜索化学结构的查找表中使用。

SMILES(http：//en. wikipedia.org/wiki/SMILES)是一种标识化合物的常见结构，其字符串编码非常紧凑，并且可以和化学结构双向转换。但是，SMILES算法有多种形式与实现，导致了同一化学实体有不同的SMILES。现在我们在此项工作中使用SMILES是由于其简单且搜索容易。正如本章的“通过唯一标识符与自描述数据格式实现数据集成”所述，将我们的SMILES转换为InChI是可行的，从而我们可以自动将数据集成到由InChI标识的数据网络中。

开放数据与可访问服务提供了广泛的可视化与分析选择

有了一个标准、自由、可访问的原始溶解度数据贮存构架，下一步就是分析数据。分析可以是简单地生成所有已完成测量的汇总，也可以是来自数据衍生模型的复杂统计描述。不论是哪种情况，都有必要通过自动化的方式访问数据以进行处理。采用其他来源的信息来丰富数据也是值得一做的。这反过来又要求主要数据需以通用的标准发布。

电子表格包含了若干列，每一行是一条测量值。溶剂和溶质都采用了两种格式表示：人类可读的通用名及机器可读的SMILES编码。由于选择GDoc作为数据的主要发布形式，为了让数据集发挥最大优势，采用机器和人类可读的两种表示是非常重要的。唯一不需要采用两种表示方式的是采用数值表示的溶解度本身。

前面已经提到，我们决定不从主数据集中剔除有问题的数值。这产

生了一个机器可读性的问题，因为并没有通行的方式来表示“这个数字有一点狡猾”。对于这项工作，我们决定在人工判读之后来标记那些被认为是不准确的数据，并给出理由。这使得任何用户在做任何分析时都可以选择采用那些记录，不管是采用人工还是自动的方式。任何进一步的详细分析都需要访问电子表格中的数据。最简单的方式就是把数据导出，比如导出为CSV文件，然后用其他软件来分析。但这样会导致无法获取到最新的数据。G公司的Doc API则可以让创建Web应用变得简单，同时又维持着与“实况”数据的连接。

作为这种应用的示例，我们创建了一个Web服务来让用户查询电子表格中储存的数据并获取表格汇总（见图16-2）。这个基于表单的查询界面为获取相关数据提供了一个快速直观的访问方式。整个页面只由HTML与JavaScript构成，不需要任何的服务器端软件。数据是通过G公司提供的API从GDoc中异步访问的。获取数据之后，就可以进行一系列的计算。在这里，我们要计算溶解度的平均值和标准差，可以用于动态高亮反常数据。不仅原始数据子集的表格输出很有用，可视化技术也可有效地概括查询结果。这个查询应用采用G公司Visualization API生成电子表格中提取数据的柱状图。从电子表格提取数据很容易，生成一系列可视化图表也是很容易的。在我们的案例中，一个显示某化合物在不同溶剂中溶解度的简单柱状图就可以提供结果的简要概括。

本应用的另外一个值得注意的地方是表格中包含化学结构的二维渲染图，渲染图是通过美国Indiana University基于REST的服务提供的。通

过在服务URL中传递SMILES代码，可以将化学结构的二维图片插入到任何网页中。同样，这些特性也不需要服务器端软件的支持。分发这种应用变得非常容易，仅仅只需要把HTML页面拷贝到另一个Web服务器上即可。

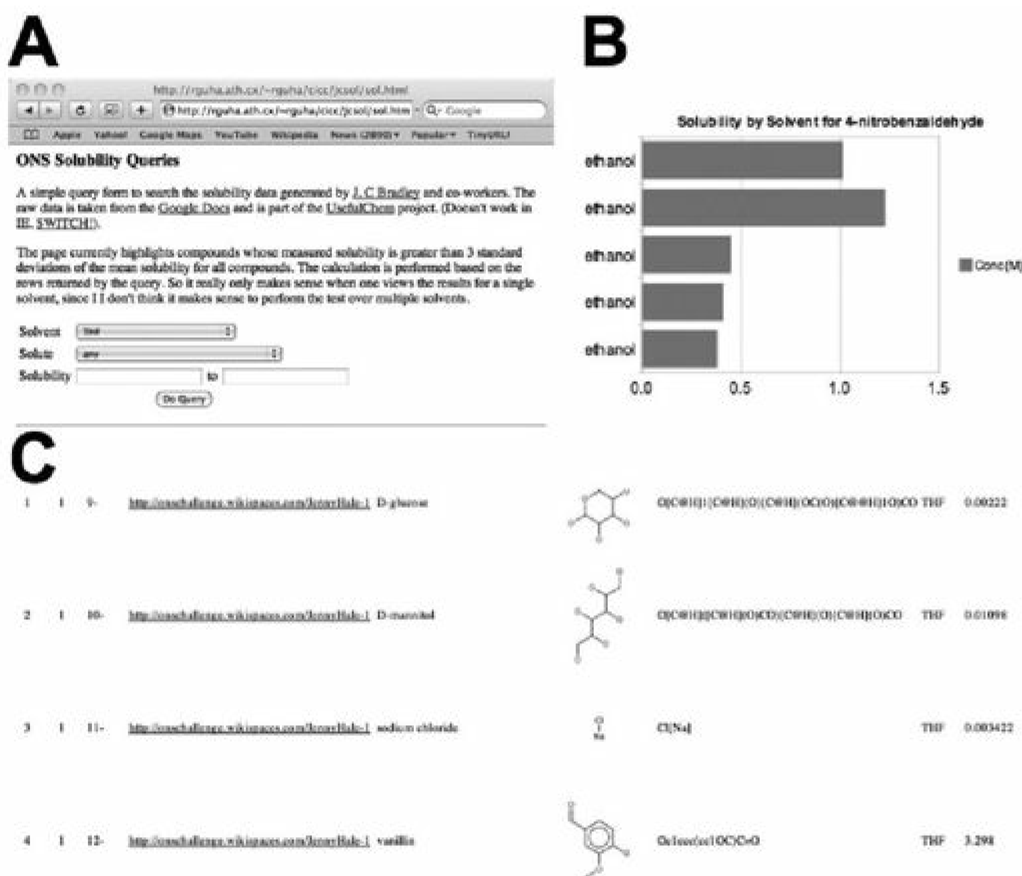


图　16-2：用于检视溶解度数据的可视化工具。A）采用JavaScript和G公司Doc API生成的一个简单表单输入界面；B）溶解度数值的图形化表示；C）带有二维化学结构渲染图的数据表格输出，可以从http：//toposome.chemistry.drexel.edu/～rguha/jcsol/sol.html访问此服务。请注意：本服务与这里提到的其他服务都是动态的，可能得到与上图所示不同的查询结果（见彩图54）

虽然这只是一个相当简单的应用程序，但它突出了这种解决方案的分布式本质，即把开放数据与具有多个来源的免费可视化方法结合起来。更重要的是，这个系统的分布式本质和可自由获得的数据使得来自不同领域的专家们——测量数据的实验员、制作接口的软件开发者以及制作创建统计模型的计算建模人员——可以更好地将他们的专业技术结合起来。开放数据、开放服务以及支持它们的生态系统的真正前途在于：这种专业技术的结合不需要建立任何正式的合作关系。研究人员可以用一种数据的测量者所想不到的方式来使用数据。不管是进行补充实验，还是做新的分析，或者提供处理数据的新服务，这样做都可以使原始的数据集更有价值，同时加强它周围的生态系统。而且不管什么时候，都可以找到指向原始记录的链接。

通过中心聚合服务集成数据

对于我们主张的方式存在一个合理反对意见，那就是：如果此方式被广泛采用，那么将会导致很多分散、不相连的数据源出现。尽管，从技术上可以利用搜索工具将这些资源汇总起来，但对于研究人员来说，这仍然是一个问题。因为他们通常习惯于使用几种他们喜欢的服务。有人校对的化学信息数据集中，美国化学会的化学文摘服务(Cemical Abstract Service,CAS)是黄金标准。CAS注册集中覆盖了超过四千万种物质(htp：//www.cas.org/newsevents/connections/derivative.html)，其中包含的数据来自出版物、专利、化合物目录以及越来越多的在线数据源，比如ChemSpider(http：//www.chemspider.com/blog/caschemspider-

connectivies-and-unintended-collaboration.html)。

ChemSpider是以“构建以结构为中心的化学家社区”为目的为化学家创建的Web资源。有来自超过150个数据源的两千余万不同的化学实体，ChemSpider已经成为了化学家寻找化学实体信息的主要Internet资源之一。每个化合物都有不同类型的信息相互关联起来，这些信息包括不同的标识符（系统命名、商品名、注册号、各语种命名），理化性质的预测值，实验获得的物理、化学与光谱数据来自一系列数据源(htp：//www.chemspider.com/DataSources.aspx)的指向链接。因此ChemSpider可以看做一个指向其他资源且基于结构的链接农场[4]。同时，ChemSpider还提供了一个操作环境，以便用户在线简要或者扩展显示数据。用户可以批注、校对数据，从而删除化学实体的错误关联，并且添加他们自己的信息，包括指向外部资源的链接或者其他数据批注。用户也可以向数据库中储存新化学结构，并为之关联光谱数据、图片，甚至是视频文件。ChemSpider中有很多种搜索功能，包括搜索预测的数据、基于结构或子结构的搜索[5]。

ChemSpider中数据主键通常是分子的通用标识符，如此就为连接其他来源的化学信息提供了一个理想的环境。ChemSpider不仅提供了化学数据搜索的中心资源，同时也是用户的数据银行，其他研究人员就可以很容易地通过它找到我们的数据。尽管一些ChemSpider关联的数据是由在线资源抓取的数据搜集的，但是对这种方式需要小心处理(htp：//www.chemspider.com/blog/care-in-nomenclature-handling-andwhy-

visual-inspection-will-remain.html)，更多的数据是在进行了某种形式的人工校对后添加的。

非水溶剂溶解度数据是作为ONS-Solubility[6]项目的一部分添加到ChemSpider中的。目前一些在线可得的数据是作为补充信息和化合物数据中可能有的其他理化数据放在一起的。溶解度数据有一个链接指向实验记录页面，这样既和ChemSpider作为指向原始数据源的链接方式一致又和我们指向原始记录的方式一致。图16-3给出了一个ChemSpider的示例。记录中有了越来越多的非水溶剂溶解度数值，可能是人工测定或者机器测定的，这些数据都会发布出来，其相应的原始来源信息也可以点击可得。

INHERENT PROPERTIES, IDENTIFIERS AND REFERENCES

2D 3D

ChemSpider ID:	62434
Empirical Formula:	$C_3H_7NO_2$
Molecular Weight:	89.0932
Nominal Mass:	89 Da
Average Mass:	89.0932 Da
Monoisotopic Mass:	89.047678 Da

H_3C O O NH_2

load save zoom

Systematic Name:	methyl 2-aminoacetate
SMILES:	O=C(OC)CN
InChI:	InChI=1/C3H7NO2/c1-6-3(5)2-4/h2,4H2,1H3
InChIKey:	KQSSATDQUYCRGS-UHFFFAOYAU
Std. InChI:	InChI=1S/C3H7NO2/c1-6-3(5)2-4/h2,4H2,1H3
Std. InChIKey:	KQSSATDQUYCRGS-UHFFFAOYSA-N

ASSOCIATED DATA SOURCES AND COMMERCIAL SUPPLIERS

SUPPLEMENTAL INFORMATION

User Data

- Experimental Physchem Properties
  Non-Aqueous Solubility: 1.32M in methanol

NAMES AND SYNONYMS

Validated by Experts, Validated by Users, Non-Validated, ~~Removed by Users~~, Redirected t

(Methoxycarbonyl)methylamine
glycine methyl ester
Glycine O-methyl ester
Glycine, methyl ester
Methyl aminoacetate

图　16-3：这是ChemSpider条目显示溶解度数据与原始数据链接的例子（见彩图55）

以后会有更多质量与校对程度参差不齐的数据在线发布，可以预想到数据提取过程会自动化。在现在的项目中我们已经在努力利用自动化过程，不过自动化过程可以推广之前有两大困难需要克服。第一个问题

是信任问题：哪些来源的数据可以给予充分的信任然后自动提取？对这些数据源应该进行何种程度的人工校对？目前对GDoc电子表格主数据的校对程度是否足够？是否还需要进一步的过滤？对目前的情况来说，我们采取的方式对一个致力于提供化学家可以无条件信任的服务还有差距。随着研究者发布更多的原始数据以及自动数据提取的增多，需要详细讨论数据应该在何时以怎样的方式发布，数据放置在上千白名单中需要怎样的标记。数据发布的认证过程会越来越多，从而为发布的数据提供质量和功能的双重标记。

第二个主要问题是功能问题：关联到ChemSpider的数据集目前来说还不多，在大多数情况下也是稳定不变的，所以人工提取过程尽管需要工作较多但是更实用一些。随着数据集的增多，这种人工过程会变得不可能。将数据作为在线电子表格发布出来对研究组来说很方便，但是并不能映射进ChemSpider或者其他中心汇总服务的数据结构中。ChemSpider是基于微软SQL Server的关系型数据库的。ChemSpidr标准Web页面中显示的理化数据是保存在数据库的数据表中的。在有数以万计数据供应者创建各种数据的世界中，需要一个通用语言来使广泛的数据重用于汇总成为可能。描述化学信息有多种相互竞争的标准。最可以信赖的方式是开发一种方法让数据尽可能以通用的形式展示。然后，就可以开发在不同描述标准间转换的服务。

通过唯一标识符与自描述数据格式实现数据集成

尽管G公司的Doc API提供了为数据集开发分析工具与可视化方法的

简单途径，但是问题仍然存在，数据的展示并不是标准格式，连唯一标识符也是如此。那些工具是根据GDoc电子表格为此数据集所写的。简单来说，它们需要人类来理解电子表格的每一列是什么描述符什么数值。尽管利用程序来判断包含SMILES代码的列是可行的，但是机器却不能知道它是溶剂还是溶质，甚至根本就不能判断到底是不是溶解度数据。要实现相互连接数据的希望（即支持ChemSpider和其他服务的自动提取），要以最通用的方式将数据提供给其他研究人员，那么就有必要提供一种遵守语法标准与描述符标准的格式。

资源描述框架(Te Resource Description Framework,RDF)，提供了将数据集以一种公认的机器可读格式发布的方式。有了这种格式，任何信息都可以转换为一条由一个“主语”、一个“谓语”和一个“值”组成的语句。举例来说，下面代码片段中所示代码表明电子表格中找到的对象叫做solute#59，在给出的URL中定义为资源。RDF使用“命名空间”，或者说采用公认概念集来定义“资源”之间的关系，资源可以是由唯一标识符指向的任何对象。这里一共有四个主命名空间。第一个是RDF命名空间本身，它定义了文件是RDF格式的，提供了诸如“由……定义”或者“是一个资源”这样的顶层概念。第二个命名空间是包含数据的电子表格，其本身也是个资源，在这里由命名空间ons定义，其中又包含了某些资源，每个单元格即是一个资源。第三个命名空间是Dublin Core(dc)，包含了诸如名称、作者和版本的概念。第四个命名空间是chem，托管在http：//rdf.openmolecules.net(RON)，用来指定某个单元格定义的分子是

被定义为一个资源的。

如前面所说，电子表格有它自己的数据结构，大体上是每一行指向一个测量值。G公司的Doc API可以让一个简单URL就可以引用一个单元格。要将此写入RDF声明中，我们需要描述单元格内容间的关系，比如“2-辛酸”和其他资源间的关系。一个简单关系指出“2-辛酸”由RON的某个资源定义，RON通过简单的HTTP URL引用。有了定义好的单元格内容，就可以使用外部资源来找出关于这个分子更多的信息。解析这个URL就可以获得RON服务定义的更多关于这个分子的RDF语句。类似地，它也可以给出SMILES和分子的名称，这些都是从电子表格中取得的。对我们数据中每条表示分子的记录都无一例外地可以定义一个标准描述让和包括系统命名、InChI和SMILES在内的其他标准定义连接起来。

```
  ＜ons:Solute rdf:about="http：//spreadsheet.google.com/
……/onto#solute59"＞
  ＜rdfs:isDefinedBy rdf:resource="http：//rdf.openmolecules.net/?
InChI=
  1/C8H14O2/c1-2-3-4-5-6-7-8（9）10/h6-7H，2-5H2，1H3，（H，9，10）"/＞
  ＜chem:inchi＞InChI=1/C8H14O2/c1-2-3-4-5-6-7-8（9）10/h6-7H，2-5H2，
1H3，
  （H，9，10）＜/chem:inchi＞
  ＜chem:smiles＞CCCCCC=CC（=O）O＜/chem:smiles＞
  ＜dc:title＞2-octenoic acid＜/dc:title＞
  ＜/ons:Solute＞
```

定义了电子表格中找到的每个化学实体之后，我们就可以将每个测量值类似上述的RDF片短表达。RDF定义了新的测量值，给出了溶剂、溶质、溶解度和测量值所属的实验。由于我们已经用化学表达定义了每

个溶质和溶剂的ID，这个测量值信息也可以被其他描述同一分子的RDF文件链接并使用。下面的RDF片段使用XML实体ons作为http：//spreadsheet.google.com/plwwufp30hfq0udnEmRD1aQ/onto#的别名来让XML可读性更好。(masurement179可以扩展为添加了“measurement179”的完整URL。）

```
<ons:Measurement RDF:about="measurement179">
<ons:solubility>0.44244235315106</ons:solubility>
<ons:solvent RDF:resource="solvent8"/>
<ons:solute RDF:resource="solute26"/>
<ons:experiment RDF:resource="experiment2"/>
</ons:Measurement>
```

这些语句，或者说三元组[7]可以用任何RDF引擎和诸如SPARQL的查询系统读取或者分析。通过使用合适的命名空间，尤其是那些被承认和分享的命名空间，可以生成基本上自描述的数据文件。我们开发了一个解析器(http：//github.com/egonw/onssolubility/tree/）来生成完整的RDF文档，可以在http：//github.com/egonw/onssolubility/tree/master/ons.solubility.RDF/ons.R下载。化学开发工具包(Te Chemistry Development Kit,CDK；参见http：//cdk.sourceforge.net/）用来从SMILES中获得包括InChI在内的分子属性。这是关键一步：将实验特定信息转换为可以有任何支持RDF系统或服务读取的数据文件。这种服务不需要理解如何处理新命名空间的特定概念，但是却可以知道如何处理这些概念所属的类别，从而可以将数据解析为使用相同命名空间的其他资源。

RDF的真正力量在于将多个资源通过链接连接起来（见图16-4）。举例来说，可以将我们的实验数据连接到DBPedia的数据中，DBPedia(http：//dbpedia.org)是一个由RDF呈现的在线信息资源。DBPedia使用名为简单知识组织系统(Smple Knowledge Organization System,SKOS；参见http：//www.w3.org/TR/skos-primer/）的命名空间来引入诸如“类别”的概念。在DBPedia中，不同的溶剂被描述为属于不同的类表，如碳水化合物或者醚类。通过将我们的数据与DBPedia中的RDF语句结合起来，可以对我们的实验数据进行诸如按照不同溶剂类别中的测量值进行查询。之所以是这样可能是因为RON中的资源将某些概念（分子的ID)与我们的数据和DBPedia中的资源链接了起来。尽管DBPedia中可能包含错误的化学命名（只要对象关联了正确的InChI即可）；尽管我们的数据没有溶剂类别的概念；尽管DBPedia对ONS命名空间一无所知，RDF还是做到了。

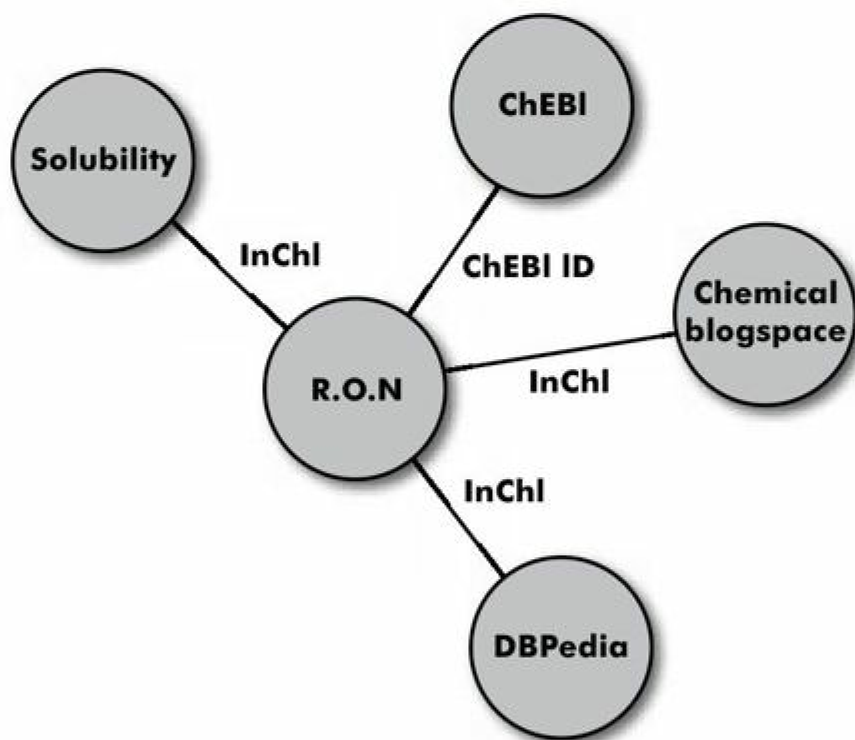


图　16-4：通过R　D　F将溶解度测量值和更广阔的数据网连接到一起。RO　　　　N(http：//rdf.openmolecules.net)即将DBPedia、Chemical Blogspace和ChEBI（欧洲生物信息学研究所的化学资源）中的记录连接到一起

将此更进一步，可以把我们的实验数据连接到Web上更广阔的天地中去，举例来说，可以利用RON中的RDF识别出讨论某个特定化合物的博客。此RDF包含了指向Chemical blogspace(http：//cb.openmolecules.net/）的链接，同时共享唯一识别符（在这里，InChI以URI的形式得到了使用）。rdf.openmolecules.net连接到了一系列数据源，这样又为不同来源的数据和分析结合在一起铺平了道路。RDF方式的价值在于可以在此图的任何一个地方插入附加数据。

共享更多共同的词汇元素可以让数据集成变得更好，不过只要有一个共同的标识符，数据集成就可以开始。

[1]CAS注册号是一种化学物质的人工编号方法，如甲苯的CAS注册号为108-88-3。

[2]化学连接表是在计算机中表示化学分子结构的一类常见格式。其基本原理为储存每个原子信息与原子之间的连接关系。

[3]IUPAC，国际纯粹与应用化学联合会。

[4]即link farm，指本身并无内容，但提供大量链接的网站。

[5]结构搜索指的是以化学拓扑结构为查询对象，采用拓扑结构匹配而非字符串查询来进行化学搜索的方式。

[6]即Open Notebook Science-Solubility，开放记录本科学溶解度项目。

[7]即包括“主语”、“谓语”和“值”的RDF语句。

结束循环：采用可视化技术启发新实验

前面说过，实验取得的数据可以采用很多种方式，从可视化到建模。这些活动都很有用，可以为手头的实际问题提供帮助。不过我们的主要目标是采用建模和分析来启发新实验的设计。随着开放来源努力的扩展，考虑可能的实验与优先度是很重要的，尤其是在最终目标是让更多感兴趣、但不一定是有经验的研究者参与进来的情况下。这种计算上的优先级在很多情况下都是有用的，如资源（经费、材料、时间）有限不能进行所有可能的实验。就溶解度来说，可能有实验员问："我们已经测试了这些化合物，下一个该做什么？"数据的可视化既可引人注意又可在资源有限的情况下为下一个实验的选择提供帮助。这就建立了实验和计算的良性循环，使得实验者和数据分析者都可以人尽其才。

要确定我们（或者别人）下一步测试哪些化合物，我们需要一种方式来了解我们所测定的化合物在化学空间[1]的位置。然后就可以确定我们的数据在化学空间中所空缺部分的位置，并用此空间内的特定分子来拟合，从而进行这些实验。这需要集成我们数据集中找不到的信息。我们有化合物和溶剂的ID也有溶解度，但是我们不知道分子的特性，也就是它们在化学空间中的位置。要获取此信息，我们需要使用一系列服务创建化学数据的混搭(mshup)。我们提供了CDK(Steinbeck 2006）描述符的一个简易REST接口。类似

http：//www.chembiogrid.org/cheminfo/rest/desc/descriptors/c1ccccc1COCC

这样的URL可以额外获取包含多个URL的XML文档，其中每个URL又指向一个含有特定描述符数值的XML文档。这里展示的可视化所采用的空间特性包括化合物分子量(M)，预测的油水分配系数(AogP，化合物亲水亲油特性的一种度量），还有计算得到的分子表面积(TSA)。Web服务中还有更多的描述符可供我们使用。

由于这里所有的服务和数据都在开放Web上提供，三方用户就可以利用这些服务和数据来进行可视化。使用GDoc中的数据和Indiana University的Web服务，我们独立开发了一个可视化工具，可以提供所有化合物在某个特定溶剂中溶解度的多维可视化(htp：//oru.edu/cccda/sl/descriptorspace/ds.php)。在图16-5中，X与Y轴都是一个特定的分子描述符，颜色表示化合物的类型，每个点的大小和颜色表示溶解度。另外，把鼠标在点上悬停可以激活一个含有更多详细信息的提示条，其中有结构和溶解度数据。此图清楚地显示了现有数据点所没有占据的化学空间部分（比如图A的左下方）。理论上讲，可以查询ChemSpider这样的数据源来给出空白部分化合物的建议。

要扩展多维展示的能力，我们准备了在Second Life[2] (htp：//www.secondlife.com；见图16-6）的3D环境中的可视化工具。和GDoc一样，Second Life也是一个古怪的科学可视化环境。不过，它一样满足了我们对项目其他部分所要求的条件。它通过一个通用的免费程序包为用户提供了一个简单（或者说相对简单）的环境。其他专门开发的可视化工具经常很复杂而且价格昂贵，和它们相比，Second Life显著

降低了门槛。从可视化体验的角度来讲，Second Life同时也有很多优点。它可以在图表周围移动，放大缩小，甚至走进图表中，从内部的角度来查看图表。不同的用户也可以同时查看和操作同一图表。从开发者的角度看，Second Life提供了将数据带到Web之外的能力，可以使用前面提到的Web服务，同时也为用户提供了一个可点击界面，可以操作图表或者随着数据点中的链接访问数据源和实验记录。从理论上来说，使用一个完全在浏览器中工作的开源渲染系统可能更好，但是开源系统十分有限，而且没有其他系统提供了像Second Life这样的技术性能、简洁的界面和可用性的组合。从实际应用上来说，这些引人注目的可视化界面已经说明了问题。

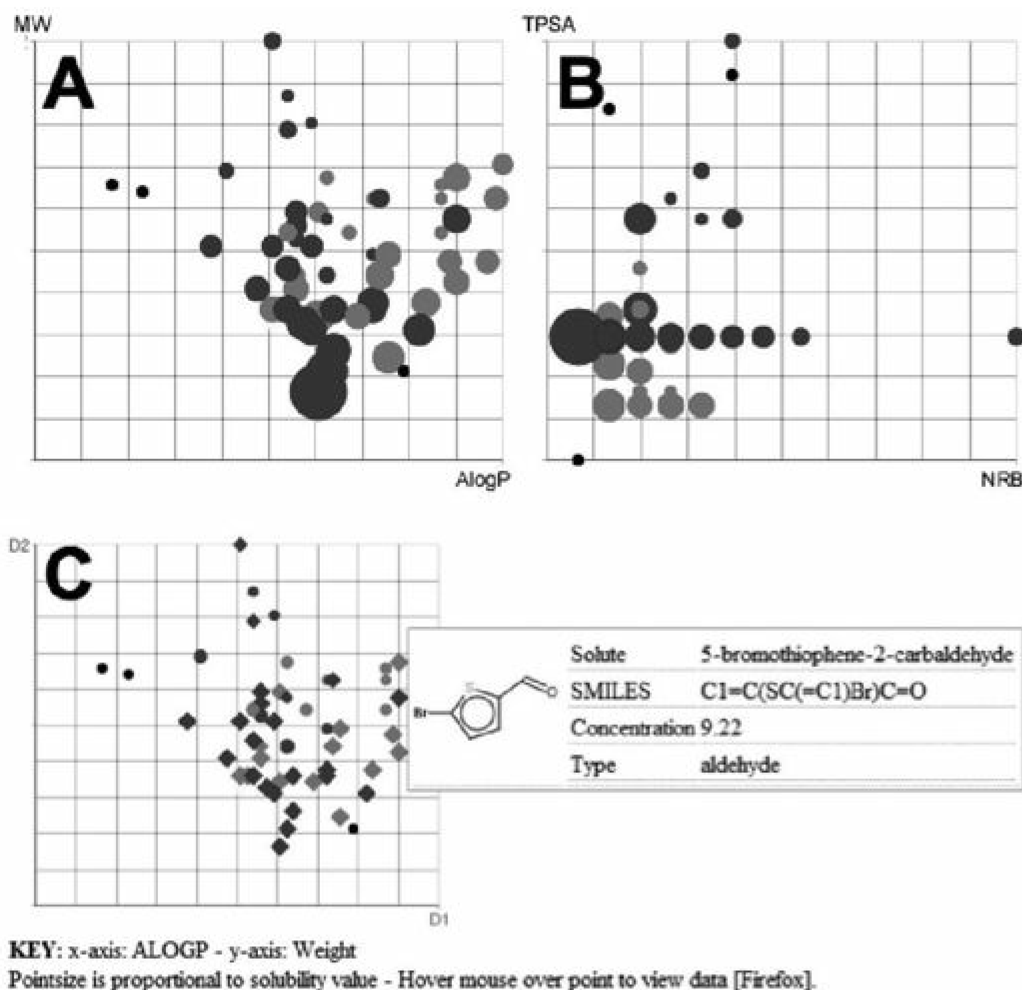


图 16-5：化学空间溶解度数据的图表展示。A和B给出了同一数据组在表示不同化学特性的数轴上的两个可视化图表。点的颜色表示化合物的类型（红色的为醛类，蓝色为羧酸类，黄色的为胺类，黑色表示其他），点的大小表示溶解度的值。表C显示了可点击的界面，其中有单个数据点的化合物结构和溶解度值（见彩图56）

很明显，易访问的数据会让计算科学家可以进行一系列分析，不过实验和计算的紧密集成则可使得整体效率更高。虽然很多计算分析需要大量的人工干预，不能转换为自动化的在线服务，不过也有很多简单的分析能转换为可以与一系列平台交互的服务。这样，分析数据来指示实

验方向、开发新的应用或者和混搭其他数据和应用都变得更简单了。这些混搭展示出了使用广为人知且易于转换、机器可读标识符的强大之处。此处的SMILES编码是关键的标识符，使用它可以从其他Web服务、数据源或者其他研究者实验数据中获取进一步的数据。在不久的将来，采用可以描述结果的RDF会极大地促进自动集成。

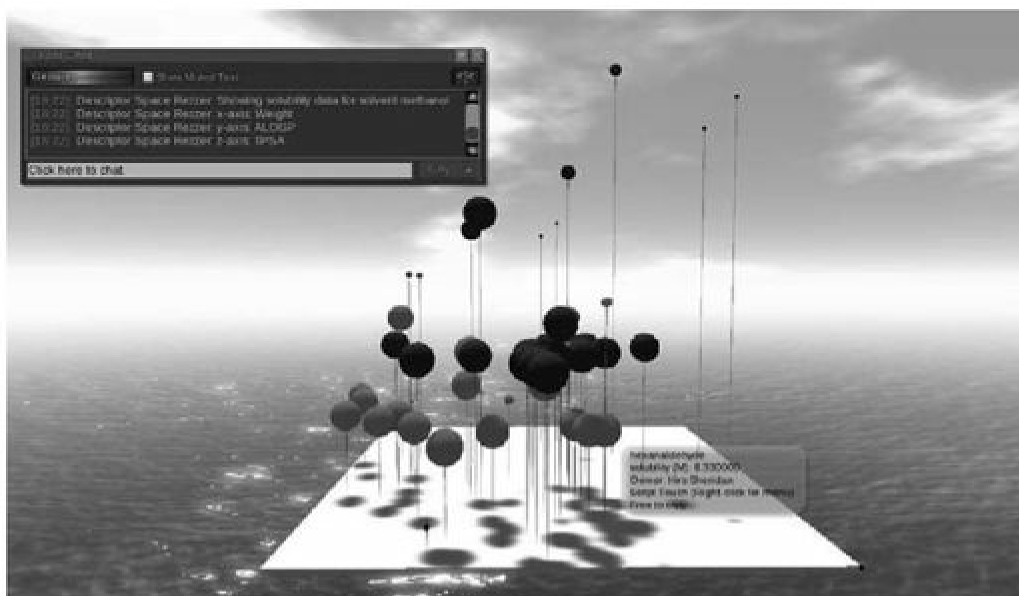


图 16-6：使用Second Life展示多维数据。三个空间轴分别表示三个化学描述符。球的颜色表示化合物的类别（和图16-5的定义一样），球的大小表示在当前溶剂中的溶解度。此可视化图表可以在http：//slurl.com/secondlife/Drexel/165/178/24找到，即Second Life中的Drexel岛（见彩图57）

由于RDF提供了一个基于公认字典的自描述框架，在不知道服务在何处且不了解内部数据结构的情况下，搜索提供感兴趣信息的数据服务也是可能的。现在大多数的混搭服务都是在已知服务、已知数据结构上，通过单个公用键（如地理位置、搜索项、日期）工作的。开放数据

Web的对象间关系都是自描述的，其真正希望在于任何人都可以创建任意的混搭服务，其中的数据搜索和信息资源都是过程的一个整体部分。

[1]即Chemical Space，指的是所有稳定化合物所组成的集合。

[2]即第二人生，是由Linden实验室开发的一个基于因特网的虚拟世界游戏。

在开放数据和免费服务下建立数据网络

科学展示和科学交流的艺术很大程度上在于设计可以剔除不准确或者是误导性结果的流程，同时又要提出证据来论证一个人们可以理解的简单解释。科学，可以看成是一个把世界的各部分简化为简单模型的过程。这个问题的一部分是将模型过分简化，而只是为了强调某一论据或者让复杂的系统易于理解。

我们的做法则是提供全部细节让人们直面真实测量的复杂性。通过在创建主数据集时采取尽可能透明的方式筛选数据记录，我们致力于在复杂性产生的问题和对清晰有用数据集的需求间保持平衡。网络存储空间几乎可以零成本获得，而且有着大量优质、免费的托管服务，这使得运行一个公开的研究记录成为了可能。这样就无法再为在文章中出现“数据不再一一列举”找理由了。不过要提供一个完整的记录又产生了几个新问题。

第一个问题是简单的体积问题。研究记录本身往往是由一系列不系统的文本和图像组成。并不存在一个由人类或者机器容易解析的通行标准。在将所记录信息转向为人所用的过程中，提取和过滤是必不可少的。我们选择GDoc电子表格作为已提取数据的主数据源。自记录中提取数据的过程，目前仍然是主观和人为的。电子表格为人类提供了一个自然的界面，在个别实验科学家看来，它同样为网络服务提供了一个有效的数据再加工和展示界面。

我们可以考虑直接从实验报告中抓取数据。使用一些通俗格式和正则表达式分析和转换，从记录的feed中读取数据自动填充电子表格是可能的。我们还没有做到这点，因为我们期待在这个阶段中有人为筛选的过程。随着项目的扩大，到了一定规模这样就不行了。在何种规模引入自动抓取将取决于项目、数据类型，以及目前呈现的数据集对与精度和准度的需求。

数据一旦公开化，它就会对任何有兴趣的研究者敞开大门，而且G公司的Doc API也使得在大范围的服务中利用数据成为了可能。这其中包括可视化或分析服务。这些服务将会依赖于对电子表格数据结构的理解，这意味着工具通常是针对某个特定数据集编写的。但是，即使在这种情况下，有效利用大范围的服务、数据源以及可视化工具来造就高效的数据展示仍是直接可行的，这些展示小到表格和简单图表，大至五维、七维甚至更多维的可点击界面。开放的标准和系统提供了把数据和信息移动到最有效之处的能力。关于真正的开放和自描述数据格式的承诺是了不起的，但是并不现实，即使是像化学这样依赖数据的科学也一样，因为以一个实验者理解的方式把记录恰当地翻译为系统的可机读格式，并要使得编码的计算机或者人类也可以理解，这是科技上和社会科学上都有的困难。这里我们已经展示了把电子表格（那些实验者所熟悉和认可的）的数据转换成RDF，不过其他的格式一样也会很简单的。

这样普遍的数据格式为开创能够综合多源数据的服务提供了可能。包含溶解度信息，或者数据集中其他信息的大范围数据源可以集中进行

综合分析。这就使得ChemSpider这样的聚合和连接农场服务成为可能，不仅仅是自动聚合数据，从技术上已经可以实现对多源数据所需校对级别进行判断，并在需要时引入人工校对。这个中心化，反过来提供了一个有价值的索引服务，而这个服务提供了一个在线的中心位置，在那里用户能够搜到他们寻找的数据。我们描述的所有工作的中心主题之一，在于免费托管系统的使用，这个系统可以在为用户避免复杂性负担的前提下提供足够多功能。对大多数的记录、聚合、分析、可视化以及展示步骤来说，这里拥有更加先进、大众或者更加尖端的可用工具。为了记录这次研究，我们本可以使用一个商业电子实验室记录本或者是一个专门设计的在线系统；然而，我们选择了免费的wiki服务。主数据的呈现可以使用一个带有内容管理系统的数据库后端来提供复杂的可视化；但我们选择了一个在线电子表格并利用其JavaScript API来展现一定范围的可视化服务。虽然有很多功能高级和复杂的三维可视化环境，但是我们选择了Second Life。

我们的部分决策理由是费用。我们使用的所有服务器都是免费可用的，并允许本质上无管理的发展过程有成长的空间，也吸引了新的低门槛的合作者。然而，此决策更多的考虑则是运用一个适合目标却不过于复杂的服务。在实验者和理论及分析型专家之间构建一个有效的连接通常都是一个挑战。将电子表格应用为数据源，不仅能够自动或人为地转换成一个有良好描述能力的格式（一个正式的关系数据库），或者是一个自描述且可扩展的格式(RF)，也能简单地直接转换为复杂的可视化，

这个应用也阐释了为何将电子表格作为结合点。实验科学家们喜欢并且理解电子表格。计算科学家们也许更倾向于文本格式，或者是可以用代码操作的数据库，类似XML和RDF的格式。把这些用户群集合在一起的关键将是具备能够自动进行格式来回转换的能力。

最后，整个工程的关键是信任和透明度。由于记录需转化为数据，数据需转化为信息，而最终信息会被转化为一个模型或者理论，在以上每一个阶段内容都有可能丢失。那些通常杂乱无章的细节，在更大的图像出现的时候总是被抛在脑后。这是完全合适的。科学，正是一个以允许人们预见未来的方式进行总结观察的过程。正如我们对服务的选择，一个科学模型或者理论如果可以做到通常用最少的时间完成实验，那么它便是有用的。然而，传统上，这个总结的过程是以无法探寻细节为代价的。在网络的世界里，存储是廉价的，在此不再需要赘述。而当前，如何做出选择，在于如何展示隐含的细节，在汇总过程中如何过滤，以及如何保持汇总结果和原始记录间的链接。

这些并不是简单的决定，我们并不会宣称我们百分之百正确。尽管如此，我们相信这个工程能够作为这项尝试的样本。四个月来，一个原本是火车上两人间的讨论已经发展成为一个跨国的数据收集、可视化以及建模尝试的项目，而这些参与者们是实实在在全部共享所有的数据和分析。如果新的研究者感兴趣，那么合作可以容易地得到发展。我们开放的数据和服务已经创建了激发兴趣的新的可视化服务，而且不需要任何实验者自身的直接参与。这些可视化不仅对于实验者是有用的，而

且，自身也是极其美丽的。不过，它们只能再现任何人利用我们发布的数据能够做到的一小部分。同时，它们通常提供通向充满瑕疵和缺陷的原始记录的链接，允许任何一个用户评定其有效性和他选择的分辨率下的任意某个数据。

美丽，通常被认为是类似简约或对称的，这是一种用简单的数学描述便可使全貌再现的感觉。而与此相比，现实中的实验数据却大相径庭。那些存在或者有时埋藏于实验数据中的美丽，或许需要大量的过滤才能浮现出来。但是，若真正的美丽存在于理解那些真正于世界最深处发生的事情之中，只要我们可以，那么我们就能够仅仅依据已知的分析来发觉那有限的一点点美丽。通过提供尽可能多的记录，我们可以让其他研究者发现和揭示更多埋藏在表面深处的美丽之路更加平坦。

致谢

作者在此感谢各位所做的努力。感谢Khalid Mirza、Jennifer Hale和Tim Bohinski完成大部分数据搜集工作。感谢Bill Hooker对开放记录本科学的帮助。感谢Submeta和Nature出版集团给予的经费帮助与大力支持。

参考文献

[1]Bradley,Jean-Claude. 2007."Open Notebook Science Using Blogs and Wikis."Available from Nature Precedings,http：//dx.doi.org/10.1038/npre.2007.39.1.

[2]Bradley,Jean-Claude et al. 2008."Optimization of the Ugi reaction using parallel synthesis and automated liquid handling."Journal of Visualized Experiments,http：//dx.doi.org/10.3791/942.

[3]Steinbeck,C. et al.2006.Current Pharmaceutical Design，12，2110-2120.

第17章　数据浅析：探索形形色色的社会定型

Brendan O'Connor和Lukas Biewald

引言

我们如何感知年龄、性别、智商和魅力？我们如何从形形色色匿名的观点中获取洞察力？去年，我和Chris Van Pelt一起构建了网站FaceStat.com，在该站点中，用户可以上传他们自己的照片，以及浏览和评判其他人的照片（见图17-1）。令人惊讶的是，该站点变得很流行。超过10万勇敢的用户上传了他们自己、朋友、亲戚、仇人的照片等，而且收集了关于比如以下几个预定问题的1000多万个评论：

·我看起来多大了？

·你觉得我看起来是否很帅？

·你觉得我和一只中等体型的狗打架能赢吗？

·用一个词描述我。

我们称这些问题为“多元的Hot-or-Not”。

心理学和社会学的研究人员已经广泛地研究了定型(sereotype)以及我们的外表如何影响别人对我们的看法。但是没有人可以访问来自如此多样化群体的如此庞大的数据。该数据比传统的实验研究要丰富得多，但是庞大的数据是否能够弥补对数据的控制力缺乏问题？实际上，真实世界的数据可能是最有启发性的：认为自己在做游戏的人比起一个在心理学入门课程上参与一个调查的大二学生会更诚实。

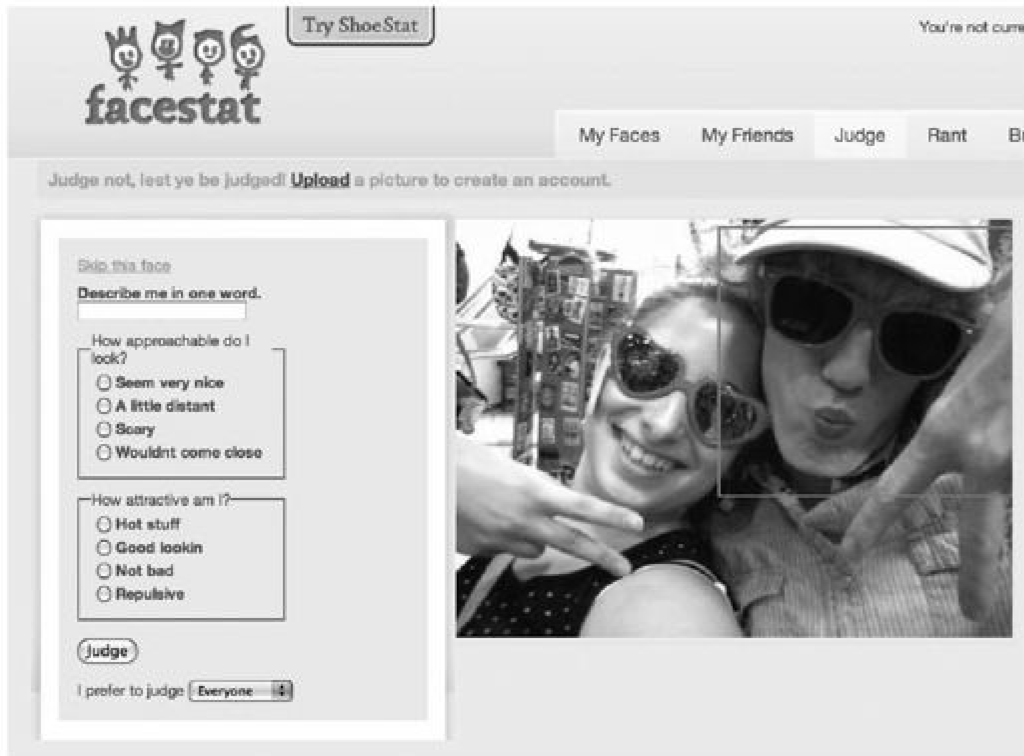


图　17-1：FaceStat的评判界面（见彩图58）

我们喜欢探索大量的数据集。我们要寻求的是有趣的模式和关联，而不是去验证预想的各种假设。我们不会试图隐瞒或者掩饰杂乱的游离点和缺失值；相反，我们将公开显示我们被迫所做出的各种选择。我们不会对种种定型做出评论或者下有争议的结论——让数据来说话吧。

预处理数据

我们从头开始：像很多网站一样，FaceStat运行在一个SQL数据库上。评判界面记录用户的评论并把它们作为一个三元组(fce ID,attribute,judgment)保存起来。我们做的第一件事是从数据库中抽取1000万条记录。它生成一个如下的文件：

```
face_id key value
149777 describe serious
18717 trustworthy 3
140467 attractive 2
149777 describe five-head
……
```

我们对探索不同类型的感知属性之间的关系感兴趣。一个有趣的问题是“我看起来多大了？”第一件要做的事是看人们给出的回复。Unix命令行工具使得快速查看人们的回复直方图变得非常简单。最常见的回复看起来是合理的年龄，但是我们也发现了一个问题：

```
Look at$cat data.tsv|
age judgments'grep"age"|
values cut-f3|
and count how many times sort|
each value occurs,uniq-c|
and order by this count.sort-nr
```

以下是这个shell管道的输出。对于每一行，第一个数字是频率计数。第二个字符串是回复值——是人们针对问题“我看起来多大了？”在Web表单中输入的回复。经常出现的情况是用户输入了一个数字，但是

有时存在一些问题：

```
7047219
7002122
6938718
6842317
……
2724\r\n
2717\r\n
2301
2116\r\n
……
1 old enough to know better
1 hopefully over 21
1 e
1??
……
```

FaceStat已经运行了8个月，而且经历了很多变化，因此数据是在不同的情况下收集的。一些奇怪的网页浏览器会增加行结束空白符\r\n。有时存在bug，用户输入文本形式的回复以及一些其他格式有问题的数据。看看在sort|uniq-c|sort-nr之后，直方图的最底部的数据值，就可以很容易地暴露出数据上存在的bug，因为它们通常会展现为游离点。我们需要写一些正则表达式，它可以清除这类格式有问题的值。

详细描述所有的数据合理性检查和数据清除会显得很繁琐，但是它们是任何数据分析重要的第一步。对于任何人工生成的数据集，必定存在一些杂乱的游离点。比如，有人发现了如何绕过选择哪张脸来评论的随机性，把一张脸标注了100多次“mr.cool”。

除了清除之外，对于该特定数据集的一些重要的决策是：1）如何把多选的回复如“很值得信任”和“不值得信任”映射为一个数值；2）如何

把很多人对一张脸的很多回复聚集成一个描述。每张脸在一些不同属性上有大约几百个评价。我们将简单地对数值评价求平均值。（在这种规范下，我们忽略文本描述，在后面会提到。）因此，每张脸包含感觉上的年龄、智商等因素的均值。它看起来如表17-1所示。

表17-1：每张脸的数据

男性	年龄	智商	魅力	警察关系
TRUE	24.26667	NA	2.800000	NA
TRUE	47.00000	3.400000	2.120000	3.2
TRUE	29.27273	2.700000	2.083333	1.8
FALSE	17.63636	3.111111	2.428571	NA
FALSE	19.58333	NA	2.750000	NA
TRUE	22.80953	NA	2.250000	NA
TRUE	29.77778	1.833333	1.900000	NA
FALSE	18.16667	NA	2.571429	NA
TRUE	46.60000	3.200000	2.120000	3.4
TRUE	52.06667	3.000000	2.080000	NA

总而言之，有成千上万的脸，包含20多种不同的属性。有很多缺失值：询问不同的人不同的问题。有了这些警告，我们可以把数据加载到某个包里进行更详细的分析。如果你想深入了解，我们已经生成了数据子集，在http：//data.doloreslabs.com可以获取有用的代码。

探索数据

有很多不错的数据分析工具。表17-2对一些最常见的工具进行了比较。

表17-2：数据分析包的比较

名字	优点	缺点	是否开源	典型用户
R	库支持，可视化	很陡的学习曲线	是	统计学
Matlab	优雅的矩阵支持，可视化	代价高，统计支持不全	否	工程
SciPy/NumPy/Matplotlib	Python，灵活通用的编程语言	各个组件集成很差	是	工程
Excel	简单、可视、灵活	大数据集、弱数值和编程支持	否	商业
SAS	很大的数据集	非常复杂，学习代价最高	否	商业
SPSS，Stata	简单的统计分析	不灵活	否	科学（生物学和社会学）

我们喜欢使用工具R，它是一个开源的统计和可视化编程环境，包含活跃的并且不断增长的开发社区。它在统计学中作为一种事实标准而产生。为了广泛的数据分析，比起其他数据分析包，我们更喜欢R工具包，因为它的图形库、便捷的索引标注以及一组优秀的复杂统计和社区维护包。你可以看看该软件，在http：//www.r-project.org可以下载，并查看一下本章最后关于它的参考资料。

R提供了很多优秀的工具来查询数据的内在涵义。通过交互式注释：

```
   Load the data>
data=read.delim（"http：//data.doloreslabs.com/face_scores.tsv"，
sep="\t"）
   and plot.>plot(data)
```

给定记录的基本表，R的默认描绘动作是为我们提供每个变量组的散点图矩阵（见图17-2）。一个很突出的方面是其年龄关联看起来很滑稽——最右一列和最下面一行。

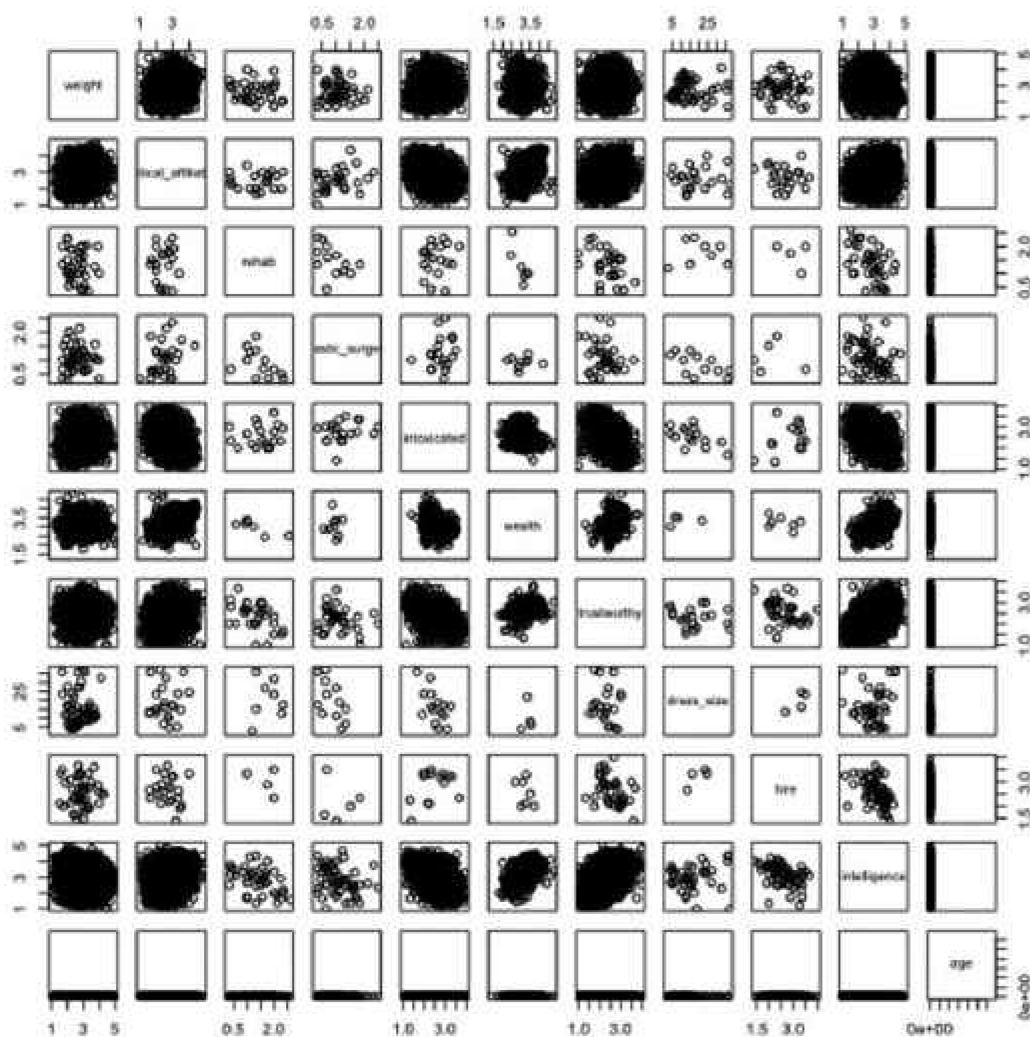


图　17-2：脸部数据的初始散点图矩阵

我们需要调查。第一件需要做的事是查看年龄值的分布（见图17-3）。

```
>hist(data$age)
```

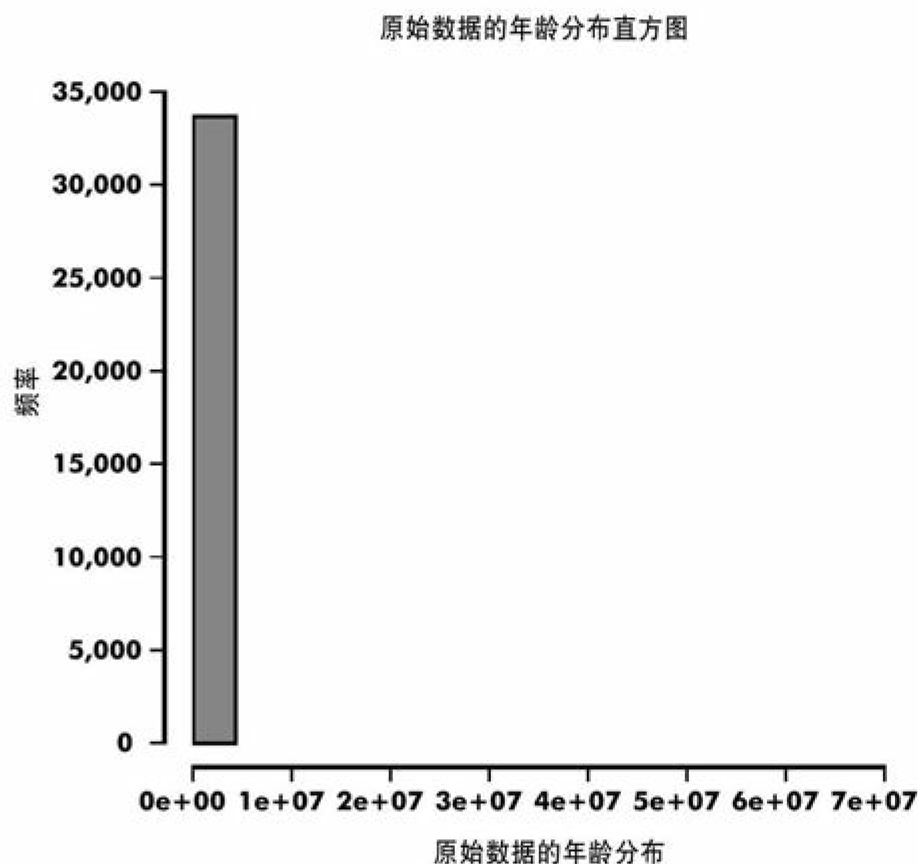
Select records with age greater than 100. > data[which(data$age > 100),]

id	评价数量	年龄	性别	魅力	智商
40623	150	402.3333	TRUE	2.416667	NA
57021	133	47882.3010	TRUE	NA	NA
66441	197	66666692.0000	TRUE	NA	NA

图　17-3：脸部数据年龄分布的初始直方图

这看起来不正常。由于存在游离点，x轴已经横向伸长到7000万。我们一起来看包含游离点的年龄值的记录：

```
  Select records with age greater than 100. ＞data[which(data$age＞
100），]
```



在初期，我们清除了非数值的年龄值，但是没有检查过分高的值。现在，最简单的操作方式是删除这些游离点。如果你之前从未使用过一

种数据分析语言，注意R的丰富的下标标注，它使得基本的探索和清除变得简单有趣：

```
  Subselect rows with age less than 100. ＞
clean_data=data[which(data$age＜100），]
```

我们再一次检查了直方图（见图17-4）发现我们的绝大多数用户是（或者看起来是）18～30岁，这看起来很合乎情理。

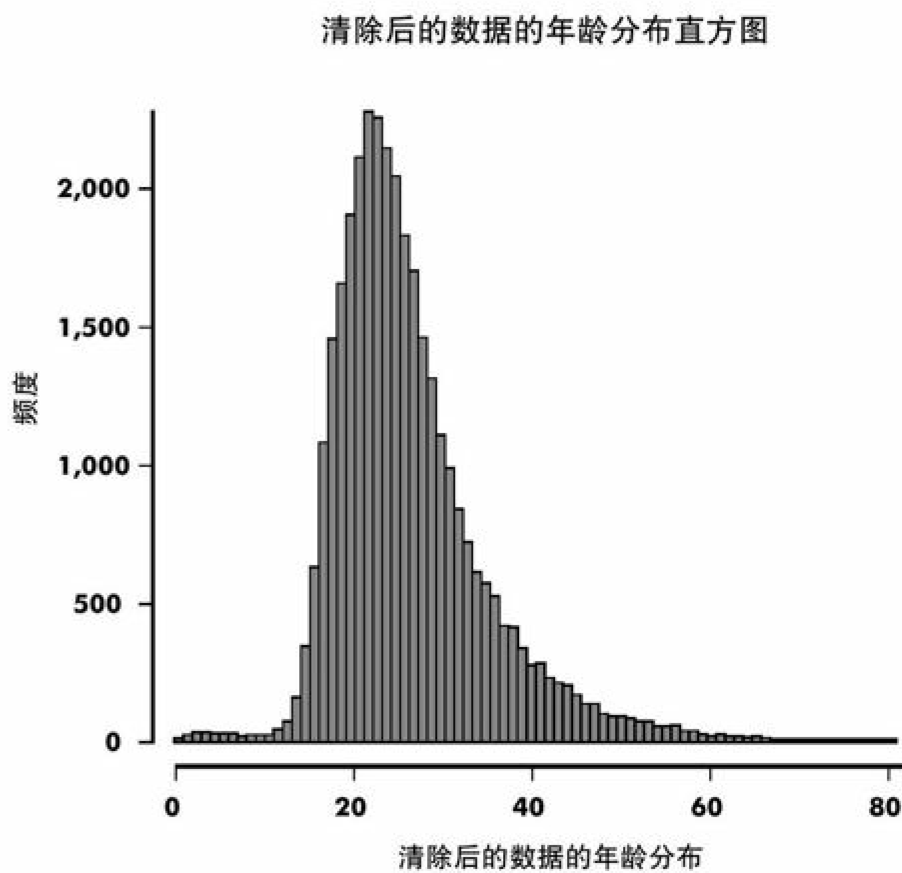


图　17-4：清除后的脸部数据的年龄分布直方图

年龄、魅力和性别

我们想要放大一些观察到的最有趣的属性：年龄、性别和魅力。每当我们用一个表来包含一些有趣的字段，把数据作为一个散点图展现出来通常是非常直观且信息量丰富（见图17-5）：

```
  Draw a scatterplot of age vs.attractiveness，＞
plot(d$age,d$attractive，
  using gender to define the
points'colors.col=ifelse(d$male，'blue'，'deeppink'））
```

该散点图包含提示作用。举个例子，女人似乎比男人更有魅力。但是很难对任何事情有一个确定的答案，因为成千上万的点描绘时互相交叠在一起。当数据过载时，散点图可能会有误导作用。处理该问题的一种方法是平滑数据，通过画估计的分布图而不是点本身（见图17-6）。我们使用称为核密度估计(krnel density estimation)的标准技术。

```
  Lay out side-by-side plots.＞par(mfrow=c（1，2））
  For males and females，＞dm=d[d$male，]；df=d[d$female，]
  draw smoothed plots，＞smoothScatter(df$age,df$attractive，
  with a color
gradient,colramp=colorRampPalette(c（"white"，"deeppink"）），
  and aligned axes.ylim=c（0，4））
  ＞smoothScatter(dm$age,dm$attractive，
  colramp=colorRampPalette(c（"white"，"blue"）），ylim=c（0，4））
```

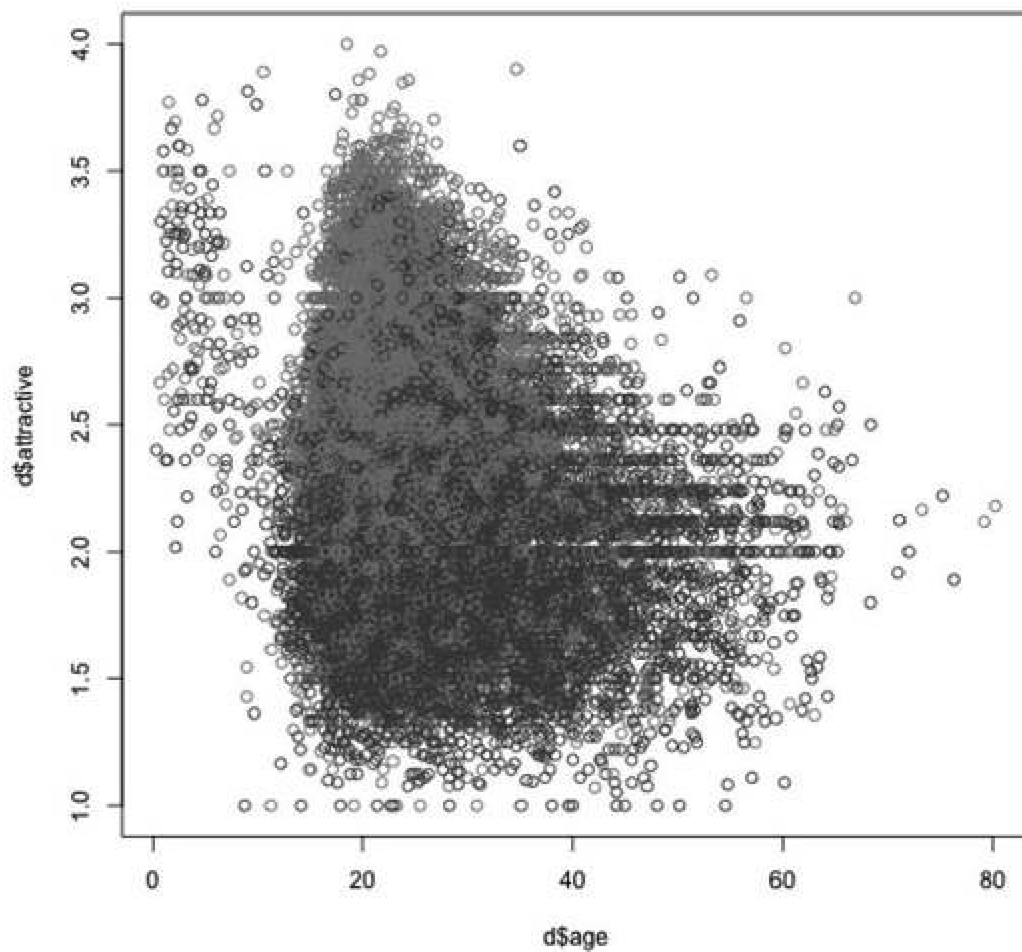


图　17-5：魅力和年龄关系的散点图，通过性别进行着色（见彩图59）

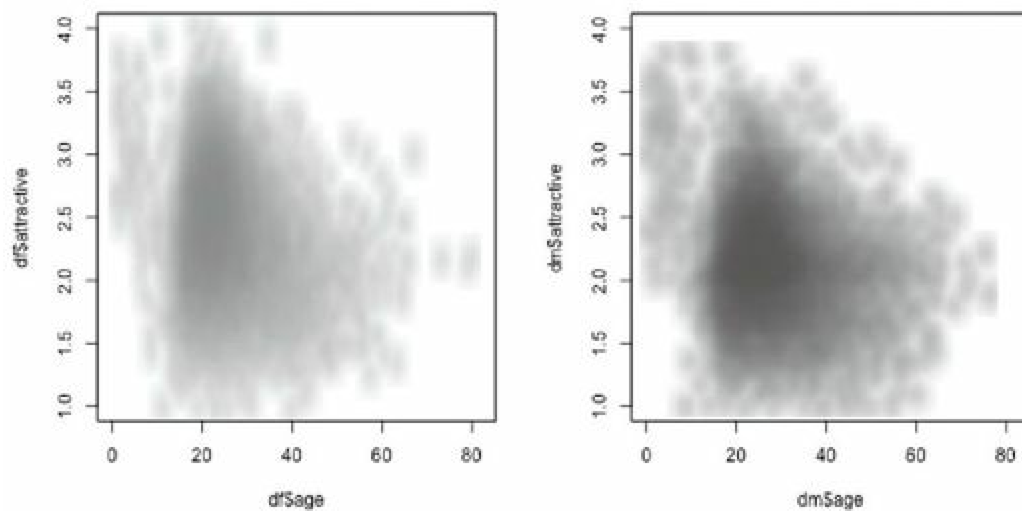


图　17-6：魅力和年龄关系的平滑散点图，每个性别一个图（见彩图

60）

我们还可以试着把它们放到同一张图上（见图17-7）：

```
> 
smoothScatterMult(d$age,d$attractive,d$male,blendFun=bl_burn,colramps=
  c(colorRampPalette(c（"white"，"red"））,
colorRampPalette(c（"white"，"blue"））,
  colorRampPalette(c（"white"，"green"））），pch=""，nrpoints=10000）
```

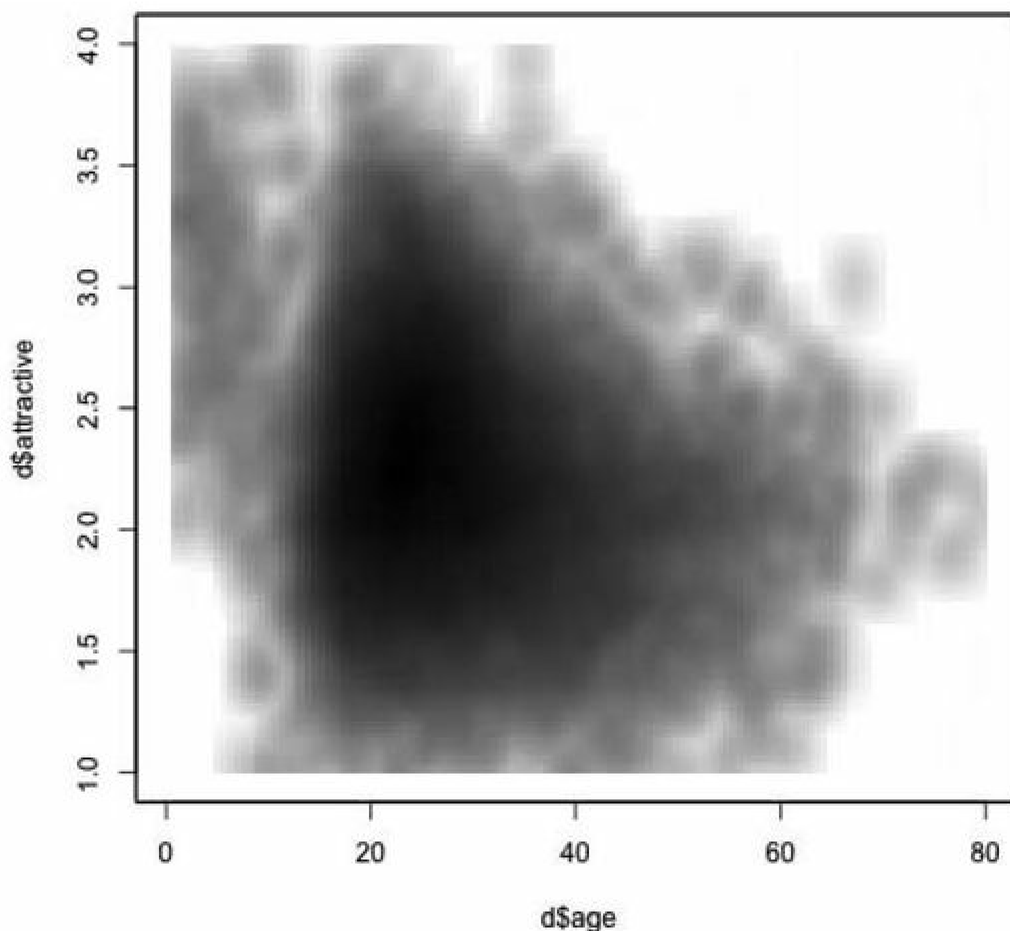


图　17-7：魅力和年龄关系的平滑散点图，通过性别着色，在一张图上交叠显示（见彩图61）

这些图形显示了数据的完全分布，但是很难看出其中的模式。举个例子，年龄如何影响魅力？通过计算总结统一和绘制它们更容易看出这

一点（见图17-8a）。

```
For males  ＞dm=d[which(d$male)，]
and females，  ＞df=d[which(d$female)，]
average across faces  ＞male_avg_by_year=by(dm$attractive，
within bins  cut(dm$age,breaks=0：80），mean)
(oe per year)  ＞female_avg_by_year=by(df$attractive，
then  cut(df$age,breaks=0：80），mean)
plot them  ＞plot(male_avg_by_year,col='blue')
all together.  ＞points(female_avg_by_year,col='deeppink')
```

这个图可以开始描述一个故事了，但还是有点难以理解。有些点是成千上万张脸当中的平均年龄值，而有些年龄更大的点来自少量的观察。因此，图的右侧有更多的噪音数据，因为样本更小。

我们给该图增加了两个特征（见图17-8b）。首先，我们计算了95%的置信区间，确保我们不是欺骗自己去看来自于噪音数据的模式。置信区间是通过我们有限的数据来估计可能的均值范围的一种方式。其次，我们将拟合局部加权回归(lcally weighted regression,LOESS)曲线，从而有助于对这种噪音序列数据的聚集模式进行可视化。通常，我们可能会对数据进行线性回归，但是该数据不是线性的，而且看起来不像我们所知道的任何一个函数。局部加权回归函数是拟合任意曲线到数据的一种方法。它基本上是执行代价高的移动平均值。该图还不够完美。图的两边有一些点包含一个或者两个可以计算置信区间的样本。如果你查看图17-4的年龄直方图，这些并不奇怪——看起来超过50岁的人们只占数据集的1.7%。而且，很多置信区间很大，以至于它们的数据点不是很有意义。因此，对于包含很少数据点的区域——特别年轻的和年老的——我

们分别使用5年和10年这两种更大的桶。该图看起来噪音数据就少多了（见图17-8c）。

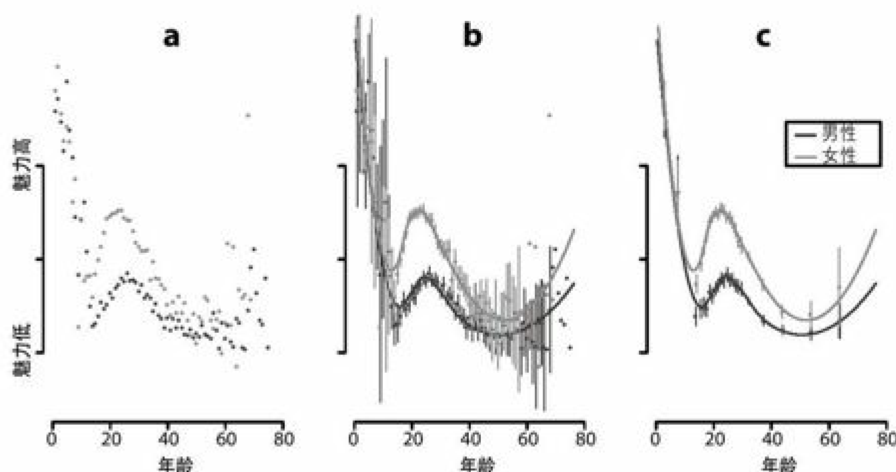


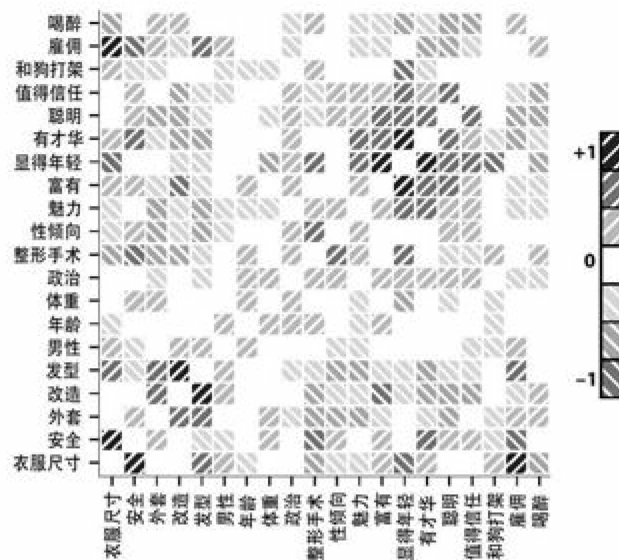
图　17-8：魅力和年龄以及性别的关系的三种迭代绘图：a）平均值在桶内区间的每个年龄；b）每个桶95%置信区间，以及局部加权回归曲线；c）更大的桶，其中数据更稀疏（见彩图62）

一般来说，在所有年龄段中，除了婴儿，女人被认为比男人更有魅力。我们发现婴儿被认为是最有吸引力（魅力）的，但是该吸引力随着年龄增长而下降，直到18岁左右（可能用户对判定青少年为“有魅力”的觉得不太自在？），然后魅力又开始上升，在27岁达到最高值。在那以后，魅力值随着年龄下降，直到50岁，在50岁年龄点，魅力值似乎又开始上升。但是很难确定，因为大于50岁的数据非常稀疏。

当然，对于20岁左右的非文本属性，存在很多关系需要探索。我们可以描绘更多类似于图17-8的图，但是我们能否马上看到所有有趣的交互？我们还是采用两两交互的方法，改变之前描绘的两两结对图。我们不是在每个面板使用散点图，而是显示单一色彩表示属性之间的全局关

联。蓝色是正向关联，而红色是负向关联（见图17-9）。

```
  First compute pairwise correlations，  ＞cors=cor(d,use='pair')
  and order the attributes to try to  ＞ord=order.hclust(cors)
  put similar attributes next to each other.  ＞cors=cors[ord,ord]
  Plot the correlation matrix，  ＞image(cors,col=col.corrgram（7）)
  with axis labels.  ＞axis（1，at=seq（0，1，length=nrow(cors)），
labels=row.names(cors)）
```



问题文本

- 衣服尺寸：我的衣服尺寸是多少？
- 安全：如果你是一名机场安检巡警，你会检查我吗？
- 外套：你喜欢我的外套吗？
- 改造：我是否需要改造？
- 发型：你喜欢我的发型吗？
- 年龄：我多大了？
- 体重：我有多重？
- 政治面貌：我的政治面貌是什么？（越高越保守）
- 整形手术：我是否做过整形手术？
- 性倾向：我的性倾向是什么？（值越高越有可能是同性恋）
- 魅力：我有多大魅力？
- 富有：我有多富有？
- 显得年轻：我是否（或者将来是否会）显得年轻？
- 有才华：我是否有才华？
- 聪明：我有多聪明？
- 值得信任：我值得信任的程度有多高？
- 和狗打架：你觉得我是否能打过一只中等大小的狗？
- 雇佣：你会雇佣我吗？
- 喝醉：我有多醉？

图　17-9：Pearson关联矩阵，蓝色方块的属性对和上升的斜线是正向关联，而红色方块的属性对和下降的斜线是逆相关（见彩图63）

该绘图蕴涵了很多有趣的关联信息，这些关系值得进一步调查：

·女人被认为比男人更聪明。

·女人被认为更有可能在和狗打架中胜出。

·衣服尺寸和体重只是弱相关。

·女人更有可能被雇佣为安全巡警。

·那些看起来做过整容手术的人被雇佣为安全巡警的概率更小。

值得信任、聪明、有才华、显得年轻、富有和保守之间都两两关联。一条“责任心·轴”？

观察标签

除了所有顺序和数值的数据，我们包含一组自由格式的标签，用户能够输入人的照片。标签范围从描述性的（“雀斑”、“鼻子带环”）到粗俗的（“抱我上床吧”、“肮脏的胸口”）到友好的（“你穿着红色的衣服很漂亮”）到建议性的（“修修头发”、“避免在太阳下暴晒”）到评论性的（“太让人惊叹了！”、“Nancy，够了！”）到卑鄙的（“那个肥胖的朋友”）到没有意义的（“……”、“plokmnjiuhbygvt fcrdxeszwaq”）。通常来说，自由格式的文本数据处理上更复杂。第一件要做的事是检查标签的分布。最常见的标签是什么？

```
  Load our tags  >face_tags=read.delim("face_tags.tsv", sep="\t",
as.is=T)
  then count  >counts=table(face_tags$tag)
  and rank them.  >sorted_counts=sort(counts,decreasing=T)
  Show the most common tags.  >sorted_counts[1: 20]
```

以下表包含了输出结果。

cute	pretty	happy	nice	fun	young
81333	40954	36263	33221	30622	27900
sweet	friendly	cool	weird	hot	gay
20362	14895	14709	12731	12662	12409
Cute	funny	scary	sexy	old	goofy
12132	11508	11445	11287	10958	10511
emo	shy				
10292	10207				

最不常见的标签是什么？

Show the least common tags. >tail(sorted_count，20)

überdude	übersöt	ünsall
1	1	1
ýour.nose.is.sexymamama!!	我	浅
1	1	1
良	?	♥haiir!!
1	1	1
白人	賢母	— —;
1	1	1
шдд	オタク	ロソリー
1	1	1
ешкув	сшеет	херня
1	1	1
ダースベイダー	Красивая!	
1	1	1

查看一些标签会对范化有问题。“cute”和“Cute”是否应该归并为相同的标签？标点符号是否应该全部丢掉？看起来很滑稽的亚洲字体的全角问号（?）是否应该认为和ASCII的问号（?）一样？显然，这取决于应用。只要可能，我们的本能是慎之又慎，保留原始数据的完整性。这保护了信息；举个例子，标签“hot”和“HOT！”肯定包含不同的语义内容。通常，当针对特定的可视化或者分析，对数据进行归并是更容易的，而不是提前猜测所有的需求，并且被强迫撤销所有值钱的范化决策。

标签分布的基本绘图查看的是标签的频率相对于其频率的排序。通常来说，当对单词或者其他词汇项进行计数时，我们看到的是从最频繁出现的单词很快下降到最不频繁出现的单词。在我们的数据，总共有240万的标签，其中有29万个唯一标签。最高的1000个唯一标签出现频率是140万——超过所有标签的一半。而在这些标签的出现频率中，有

非常显著的下降趋势。从常用标签表中，我们看到最常见的标签“cute”出现了3.6万次，而第二常见的标签“pretty”出现的频率只有它的一半。

```
For the top 1，000 tags，   ＞s=sorted_counts[1: 1000]
draw a plot of their counts.  ＞barplot(s)
```

1935年，语言学家George Zipf观察到词频分布遵循“幂函数定律”(pwer law)，其中第n个单词的频率和（1/ns)成正比，其中s是个常量。和高斯分布不同，该分布包含有限的变量，这使得它不适合于某些统计算法。流行的书如Nassim Nicholas Taleb的《The Black Swan》(Rndom House出版）和Chris Anderson的《The Long Tail》(Hperion出版）使得这些分布分别以“重尾”和“长尾”分布而著名。实际上，我们的数据包含一条很长的长尾：22万个单词，或者词汇的76%只出现一次。

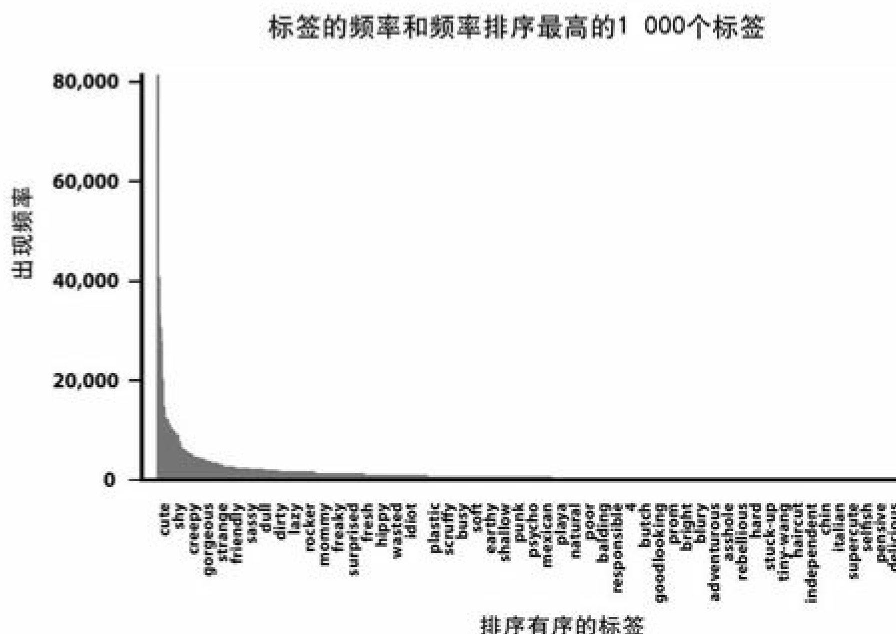


图 17-10：出现频率最高的1000个标签的标签频率

通过在对数空间描绘我们的单词的出现频率图，我们可以查看这些单词出现的频率是否满足幂函数定律分布（见图17-11）：

```
Plot log ranks  >log_ranks=log（1: length(sorted_counts)）
against log frequency.  >plot(log_ranks,log(sorted_counts)）
```

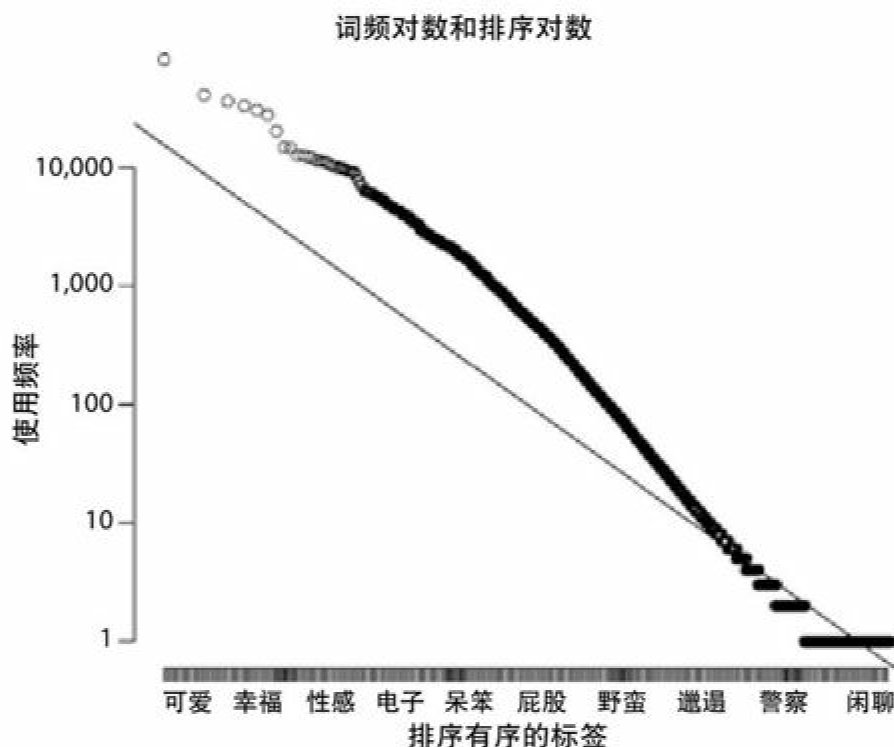


图　17-11：通过对数排序的标签的对数频率，包含幂函数定律模型的拟合线

幂函数定律分布在对数-对数空间上，看起来应该是线性的：

```
Fit a model of log count against log rank  >
model=lm(log(sorted_counts)~log_ranks)
and draw it on Figure 17-10.  >abline(model)
```

我们发现标签的概率很接近（$1/n^{80}$）分布。（如果你觉得它看起来像最佳拟合线，记住所有点的76%是在数据的最右下侧的边上）

如果你在任何文本——报纸、小说、Web页面等上描绘这种对数-对数频率的绘图——它看起来都很相似[1]。当然，当FaceStat的用户写描述标签时，它们参与的是一种语言行为，这种行为和其他类型的人类通信在本质上很相似。

这些标签如何拟合剩余的数据？第一步需要做的是从标签中随机选取样本，在我们已经生成的绘图上绘制（见图17-12）。

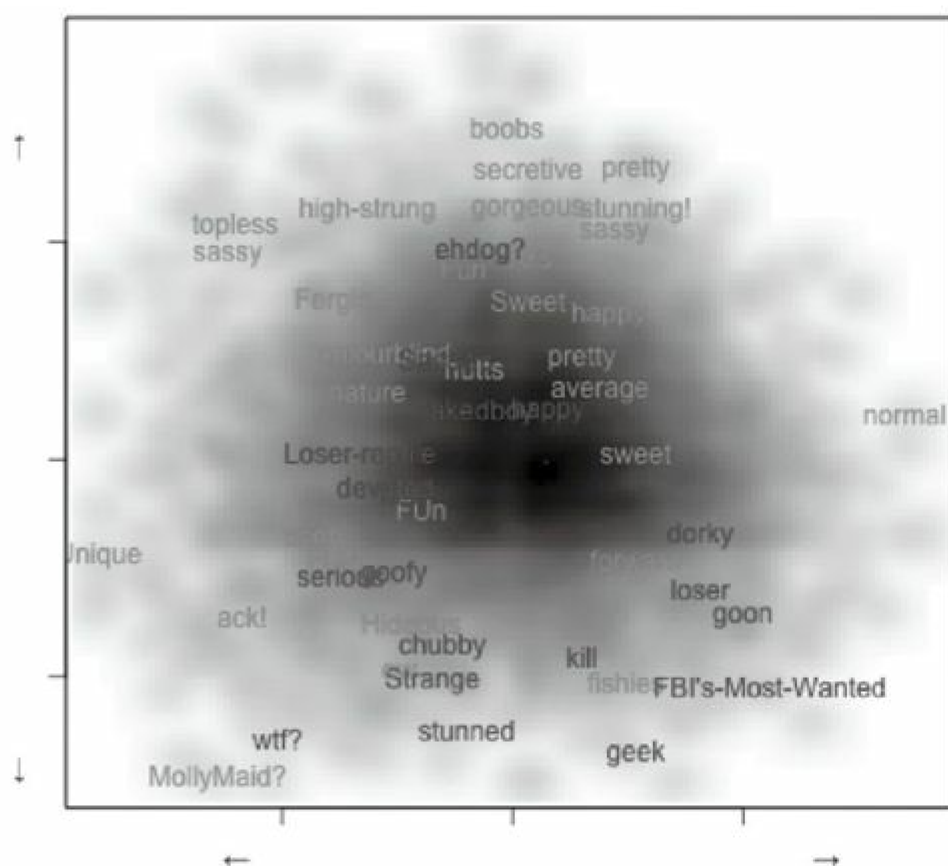


图 17-12：基于平滑后的魅力与年龄描绘的标签样本图（见彩图64）

在该图中，深暗色表示政治背景和魅力的密度的全局分布。单词是从分布中随机抽样出来的。蓝色词表示男性标签，粉色词表示女性标

签。这么做可以使我们感觉到标签是否和图中的变量一致。数据看起来很合理，标签“平均”显示在图形的中央，而被标记为“接近顶尖”的人显示在“自由/魅力”象限，而标记为“土里土气的”则是在“保守/丑陋”象限。该图可以通过不同的随机种子生成多次，可以查看全局数据的标签分布。

[1]Zipf,George.1935.《The Psychobiology of Language》(MT出版社）参见http：//en.wikipedia.org/wiki/Zipf's_law。

哪些单词具有性别化

很多社会理论学家考虑性别在语言中的不同显示的程度如何。我们的数据集使得可以从单词层次探索这一点：我们可以发现哪个标签是男性脸孔或者女性脸孔的最显著的特征。我们可以统计出对于男性出现最频繁的单词，以及对于女性出现最频繁的单词，但通常来说，这只能获取在任意地方出现频率都很高的单词。一种更好的方式是通过对性别间出现频率的标签进行打分。也就是说，为了确定标签T表示性别G的特征性，看以下表达式：

$$\frac{\text{性别是}G\text{的脸的标签}T\text{的出现次数}}{\text{标签}T\text{出现的总次数}}$$

这个表达式有个缺陷：出现频率很低的标签会带来噪音数据。举个例子，任何只出现一次的标签，它表示的性别就得到满分1（这是由于样本数据量小造成的错误，我们称之为稀疏年龄桶）。解决该问题的一个简单的方法是设置词频阈值。在这种情况下，我们只查看出现次数大于100次的标签。

计算这些分值——用统计学术语来说，它们是条件概率Pr(G|T）的最大可能估计——我们可以获得如下一些最大可能估计值表。

最具有男性特征的单词如下表所示。

	G	T	Ratio
daddy	122	122	1.0000000
fatherly	115	115	1.0000000
fratboy	177	177	1.0000000
father	172	173	0.9942197
dad	341	343	0.9941691
douche	229	231	0.9913420
Handsome	110	111	0.9909910
scruffy	149	151	0.9867550
bald	343	350	0.9800000

（续）

	G	T	Ratio
jock	395	404	0.9777228
handsome	510	524	0.9732824
thug	141	145	0.9724138
tool	255	264	0.9659091
player	522	542	0.9630996
Gay	307	319	0.9623824
jerk	131	137	0.9562044
gamer	103	108	0.9537037
fag	148	156	0.9487179
pimp	121	128	0.9453125

最具有女性特征的单词如下表所示：

	G	T	Ratio
Bubbly	118	118	1.0000000
Mom	161	161	1.0000000
busty	148	148	1.0000000
milf	267	267	1.0000000
mom	1,088	1,088	1.0000000
motherly	396	396	1.0000000
partygirl	221	221	1.0000000
mommy	307	308	0.9967532
mother	358	360	0.9944444
ditzy	144	145	0.9931034
fjortis	113	114	0.9912281
MILF	103	104	0.9903846
Pretty	926	935	0.9903743
cheerleader	159	161	0.9875776
boobs	153	155	0.9870968
makeup	143	145	0.9862069
bitchy	284	288	0.9861111
cougar	141	143	0.9860140
slutty	538	546	0.9853480
slut	509	517	0.9845261

可能令人惊讶的是这些单词如“handsome”（英俊）、“gamer”（赌徒）、“Bubbly”（活泼的）和“slut”（骚货）是多么地具有性别化特征。它们几乎总是和它们所表示的性别一起出现。

聚类

数据中的人们有哪些典型类型？聚类是找到这种模式的强大的统计方法。聚类算法通过将相似的实例分组在一起，从而把数据点分成几种特征类。聚类有很多方法，但是最流行和简单的方法称为K均值聚类（K-means)方法。在K均值聚类方法中，每个聚合有一个中心点，称为“质心”(cntroid)。在数据中找到一些不同的质心，每个数据点被分配给一个质心。该算法迭代调整聚合，这样使得尽可能多的数据点靠近分配给它们的质心。

在我们的数据集中，每张脸大约有20个数值属性。因此，脸部就是在20维空间的点。K均值聚类方法会把脸孔放到该空间的不同的聚合，试着选择聚合，使得脸孔可以尽可能地和它们所在的聚合的中央相似。

K均值聚类算法不好的一点是你必须选定固定值的聚合个数——“K”。然而，不存在很明显的方式来选择聚合的个数。需要做的最好的事是尝试集中不同的个数，查看出现了什么模式。以下是运行K均值分类算法的一个合理输出：

```
Preprocess the data, >norm_data=apply(d, 2, function(x){
by changing missing values to the
mean,x[is.na(x)]=mean(x,na.rm=TRUE)
and unit-normalizing values,x=（x-mean(x)）/sd(x)
which usually makes k-means work better.x})
Then run k-means for 5 clusters, >clus=kmeans(norm_data, 5)
and plot attractiveness vs.age, >plot(d$age,d$attractive,
but color by col=c（"red", "purple", "blue", "orange",
cluster assignment, "green", "darkturquoise"）[clus$cluster],
```

```
and have fun with unicode.pch=ifelse(d$male, '\u2642', '\u2640'))
```

散点图的点表示脸部，点的颜色和它们被分配的聚合一致。我们将显示在魅力和年龄空间内的脸庞，如我们之前的绘图一样。一些聚合已经可以理解：橙色聚合表示年长的人，紫色看起来是有魅力的年轻人等。

这个绘图只显示数据的二维或者三维，因此没有充分地总结聚类算法，聚类算法完整地比较了20维空间的人脸。这是一些聚类重叠(oerlap)的原因：举个例子，红色和绿色看起来有非常相似的年龄和魅力范围。这些聚合必须通过其他属性区分。

我们通过各种方式来分别查看各个聚合。首先，我们显示聚合的属性权重。它是聚合的质心点的位置，质心点通常被认为是该聚合表示人脸的典型属性。因此，如果你查看图17-13的每个集群的平均点，它会返回年龄和魅力的聚合权重。（我们将显示八个属性；其他属性不重要，因为它们包含太多的缺失值。）其次，我们显示了在该集群中的人脸的最显著的10个特征，通过如之前的性别分析那样的条件概率进行排序。

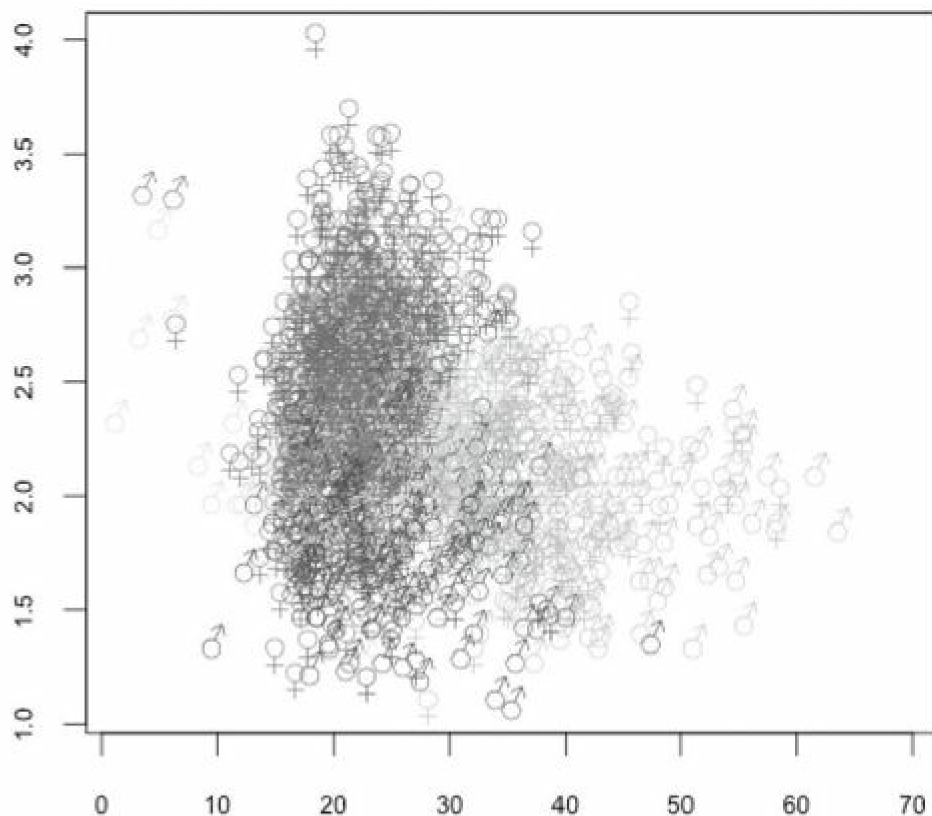


图　17-13：魅力和年龄的关系，通过集群着色，显示了包含2000个点的子样本（见彩图65）

首先，图17-14显示的是紫色集群。这是一个非常女性化、具有高度魅力值的集群。标签很有趣。它们和属性惊人的相似；事实上，如果覆盖左侧图，你很可能猜出该组的很多属性。标签显示了一致生动的画面——即使我们的k均值聚类算法完全忽略了这个信息！这说明了标签和社会属性有内在的联系。（这一点可能不足为怪。）

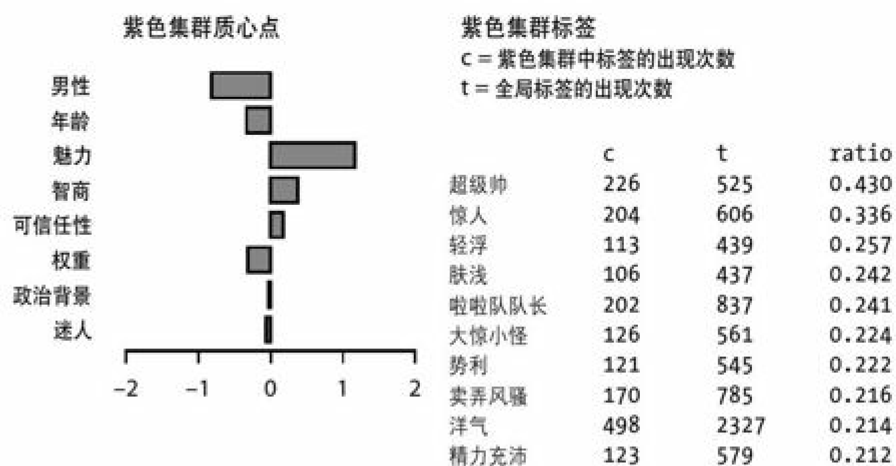


图 17-14：集群2

我们在图17-15和图17-16展示了所有集群，包括每个集群中四个最有代表性的脸孔（表示最接近质心点的）。

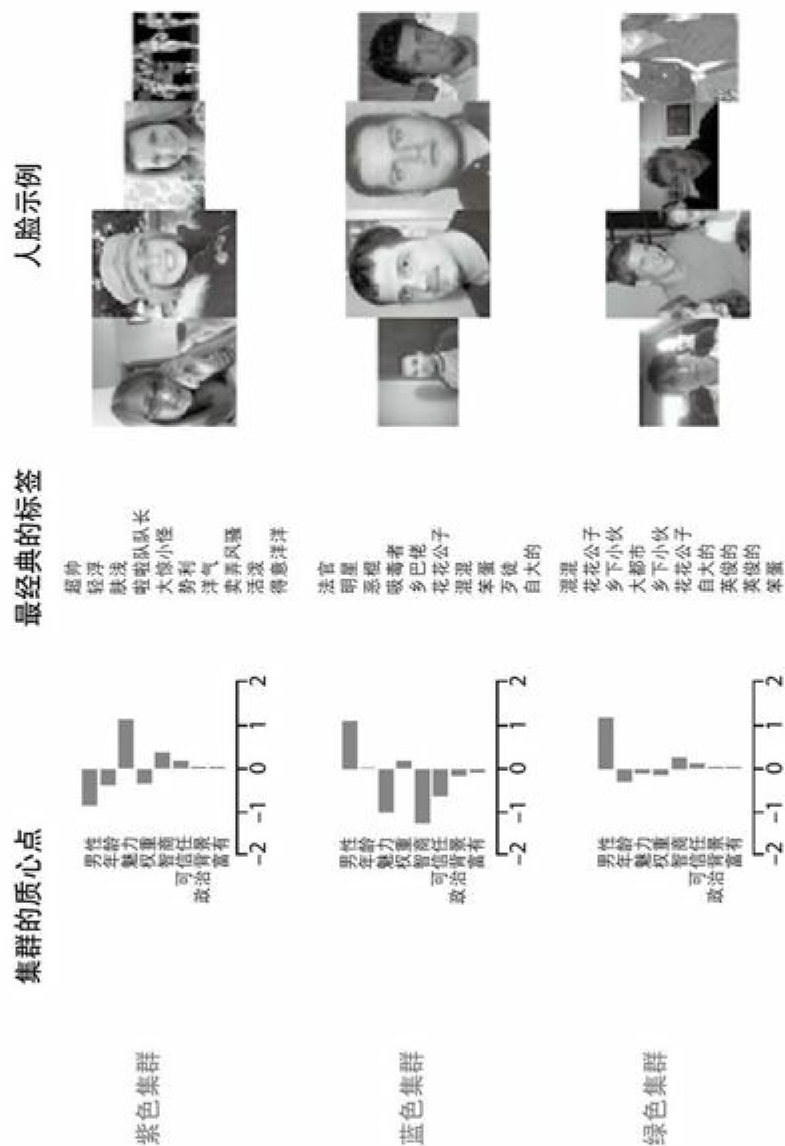


图　17-15：集群的质心点、标签和例子（见彩图66）

有些集群有直接的解释，而另一些则不是很清晰：

紫色集群

年轻、有魅力的女性。

蓝色集群

丑陋、智商低的男人（“失败者”？）。

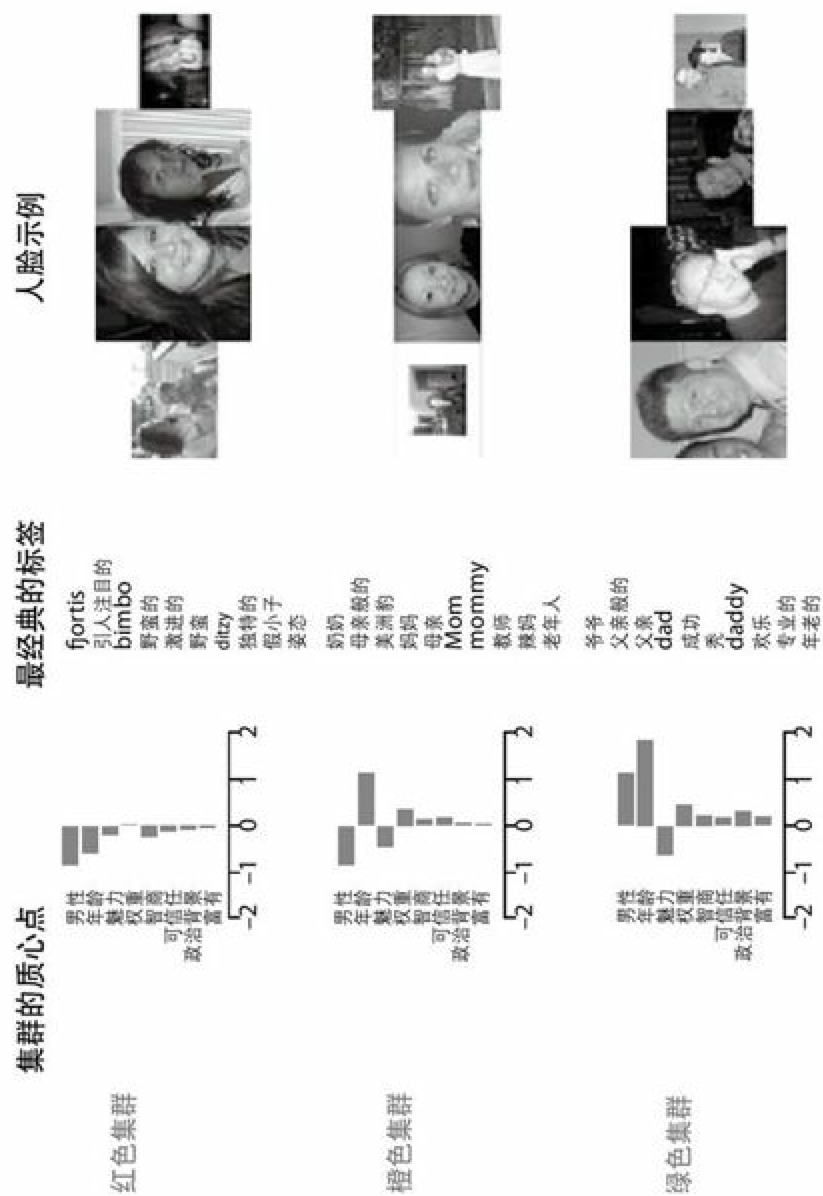


图　17-16：集群的质心点、标签和例子（见彩图67）

绿色集群

其他更普通的年轻男人。它的很多标签对于蓝色集群也非常有可能。

红色集群

其他年女性。

橙色集群

较年老的女性。

青绿色集群

较年老的男性。

聚类一方面用于发现数据高纬度的模式或者分组会很有用，因为这些模式和分组无法通过两个维度进行可视化。另一方面，很难验证聚类是否告诉你一些“真实”的东西。存在很多聚类算法和很多参数调整（比如k），不同的调整会生成不同的结果。这项工作是否有用？

集群看起来似乎相当一致，而且对于很多模式具有提示作用。观看那些生动的标签集合互相关联很有趣。而k均值聚类算法可能可以模拟我们的大脑中对人类特征的思考。从质心点和标签集合，我们可以想象表示每个集群的典型的人类特征。

结论

数据表明，一些人持有一些常见的“偏见”，认为女性比男性更有魅力。年龄对于魅力的影响，女性比男性高。社会属性的空间和我们所熟悉的很一致：乡下小伙、父辈、有魅力的年轻女性。但还是有一些潜在的令人惊讶的地方：婴儿最有吸引力、看起来保守的智商更高等。我们还发现一些性别化单词的例子。

我们试着一直继续探索这些有提示性的发现，但是本章不会给出任何特定的结论。相反地，我们想要显示在关于人们的判断的大量、混杂的数据中包含的有意义的模式的富集合的一些例子。更严格的数据收集处理（如谨慎控制的实验）将永远都无法生成如此大量的数据，因此对于后续的实验没有太大作用。

每天我们都通过自己购买的东西、使用的Web站点、搜索的查询、发送的消息和去的地方方方面面地展示自己。不论我们喜欢与否，在人类历史上，这是第一次把所有的数据都精心保存起来。暂不考虑重要的隐私问题，该数据对于社会科学的价值是非常巨大的。通过这种改变用途的信息的混乱，我们将以全新的方式了解自己。

致谢

非常感谢给本章初稿反馈的所有人：Joanna Gubman、Sasha Goodman、Jeff Hammerbacher、Mike Love、Will Moffat和Toby Segaran。FaceStat的存在归功于工作努力且杰出的同事Chris Van Pelt。

参考文献

我们上传了部分数据子集以及一些笔记和代码来帮助重现我们的分析，在：http：//data.doloreslabs.com。

如果你对学习建模工具R感兴趣，我们推荐两个Web站点：

Quick-R(http：//statmethods. net)

Robert Kabacoff写的高级概要和主题指南。

RSeek(http：//rseek. org)

Sasha Goodman实现的R文档、包和邮件列表的搜索引擎。

统计工具R的官方Web站点可以通过http：//www.r-project.org访问。如果你对它和其他的数据分析包比较感兴趣，请参见表17-2的很多评论，可以通过http：//anyall.org/blog/?p=421访问。

最受推荐的学习R的书是Peter Dalgaard的《Introductory Statistics with R》(Sringer出版，2008）。

除了统计工具R的核心功能，我们使用的一些附加包包括corrgram、flowCore、gclus、geneplotter、plyr和pixmap。

Trevor Hastie、Robert Tibshirani和Jerome Friedman合著的《The Elements of Statistical Learning》(Sringer出版，2008）包含了聚类、局部加权回归法的优秀的概要总结以及其他机器学习技术。

关于标签部分仅仅涉及统计语言分析最基础的部分。为了了解更多信息，可以参考Christopher Manning和Hinrich Schütze著的《Foundations

of Statistical Natural Language Processing》语料语言学一章(MT出版，1999），以及Daniel Jurafsky和James H.Martin著的《Speech and Language Processing》(Pentice Hall出版社，2008）。

对于魅力和年龄的变化关系分析，存在很多更好的方式来估计置信区间。一种方法是部分集中(prtial pooling)；参考Andrew Gelman和Jennifer Hill的《Data Analysis Using Regression and Multilevel/Hierarchical Models》（剑桥大学出版社，2006）的252～258页。我们在本章所做的称为“探索性数据分析”(eploratory data analysis,EDA)——和那些通常在统计方法学课程上教的精心实践的假设检验不同。探索性数据分析被统计学家John Tukey在1977年以之命名（译注：即《Exploratory Data Analysis》）(Adison-Wesley出版）的书中广泛推崇。

我们的创业公司Dolores Labs专注于外包项目：从大量的人中收集人类的任务数据来解决内容控制、信息抽取、Web搜索相关性以及其他领域的实际问题。我们收集、查看并自动分析大量关于人们的判断的数据。你可以查看本章的后续，关于其他主题如性别、颜色和宗教的分析，在博客http：//blog.doloreslabs.com可以获得。

第18章　旧金山海湾区之殇：次贷危机的影响

HadleyWickham、Deborah F. Swayne和David Poole

引言

房地产市场在过去几年得到了媒体的极大关注。大约从2000～2006年，随着价格的飙升，我们既惊奇又焦虑地看着房地产行业。从那时开始，随着信用成为稀缺以及丧失抵押品赎回权的蔓延，房地产业也跌入了冬天。在本章中，我们通过分析从2003～2008年旧金山海湾地区50万的住宅销售来近距离探讨这个“故事”。从一个地区的价格在一个很大价格范围内上升和下降的方式，我们可以学到些什么呢？

我们将从描述数据、如何获取数据以及如何通过重构、转化、清除和增强提升原始数据开始，从而为数据分析做准备。随着分析逐渐深入，我们的大多数发现都会通过图像形式展示。在此过程中，还会介绍一下我们使用的一些工具，这些工具多数都是可以免费获取的。我们的主要工具是R，它提供了统计编程和数据分析环境，在数据分析的各个阶段中都用到了这个工具：获取、清除、分析、诊断和表示。

我们是如何获取数据的

在我们确定对房地产的实际销售数据感兴趣之后，数据搜索过程马上随之展开。数据搜索并不总是成功的，因此当我们发现了《San Francisco Chronicle》杂志生成的每周海湾地区(By Area)的住宅房地产（住房、公寓、托管公寓）的销售情况（在http：//www.sfgate.com/homesales/可访问）之后，我们觉得特别的幸运。当我们得知自己不需要通过解析Web页面来抽取数据，而是已经可以获取计算机可读格式的数据时，我们感觉自己更幸运了。

每个人可读的(HML Web页面）每周总结都是基于如下的文本文件构建的：

```
rowid: 1
county:Alameda County
city:Alameda
newcity: 1
zip: 94501
street: 1220 Broadway
price: $509，000
br: 4
lsqft: 4420
bsqft: 1834
year: 1910
```

每个星期的数据可以使用如下模式的URL获取：http：//www.sfgate.com/c/a/＜year＞/＜month＞/＜day＞/REHS.tbl。这非常方便，因为只需要生成一个从他们开始记录的日期2003/04/27（我们

从他们网站的归档页面中找到的）到最近2008/11/16（截止到分析时间）的所有星期天的日期列表。有了这个日期列表，我们生成正确格式的URL列表，通过Unix命令行工具Wget就可以下载它们了。我们之所以选用wget是因为它支持断点续传功能，这样当下载中断时，可以很容易地从上一次下载到的位置继续开始而不是从头开始。

所有的数据都保存在本地计算机上，下一步是把数据转换成标准格式。我们通常使用csv（逗号分隔值，comma-separated values)格式；csv文件很容易生成，而且为各种统计包（包括Excel！）所支持。我们生成了一个csv格式的文件，如下：

```
  county,city,zip,street,price,br,lsqft,bsqft,year,date,datesold
  Alameda County,Alameda，94501，1220 Broadway，509000，4，4420，1834，
1910，2003-04-27，NA
  Alameda County,Alameda，94501，429 Fair Haven Road，504000，4，
6300，1411，1964，2003-04-
  27，NA
  Alameda County,Alameda，94501，2804 Fernside Boulevard，526000，2，
4000，1272，1941，2003-
  04-27，NA
  Alameda County,Alameda，94501，1316 Grove Street，637000，3，2700，
1168，1910，2003-04-
  27，NA
```

可能原始的数据格式更适合人们阅读，但是这种csv格式更适合计算机处理。它不但更标准，而且更紧凑（c s v格式的数据大小是45M B，而原始数据格式的数据大小是90MB)。如果你仔细查看样本数据，你可能会注意到有些东西需要解释：NAs。NA代表“不适用”(Nt Applicable)，而标记值R用于表示缺失值。在分析中我们必须考虑缺失值。

解析所有293周的数据并创建结果文件house-sales.csv只需要花费几分钟的时间，该csv文件包含521726个观察和11组变量。调整解析器使得它满足所有的边界情况，这个过程花费了更多的时间：我们需要把价格格式的数据转换成常用的数字（通过删除“$”和点“.”符号），把时间解析成一致的格式，而且为没有在所有的表都出现的字段填充缺失值。

地理编码

当第一次查看数据时，我们认为对所有的436106个唯一地址进行地理编码是非常重要的。也就是说，我们想要为每一个地址关联一个经度和纬度值，这样可以很容易地探索细粒度的空间效果。这是一个很有趣的挑战：你如何对将近50万的地址进行地理编码呢？

我们从查看G公司和Yahoo！提供的Web服务开始。这些服务不适合有两个原因：一是它们对于每天的请求个数有严格的限制；二是使用结果数据时存在繁琐的限制约束。单单请求限制就意味着对所有的地址进行地理编码可能需要花费1个多月的时间，而且其授权将会影响到公开发布结果！进一步调研之后，我们发现一项非常有用的开源服务，USC WebGIS，它是由南加州大学的GIS研究实验室提供的(Gldberg和Wilson 2008）。该服务对于非商业使用是免费的，而且对于结果数据的使用没有什么约束。我们开始使用该服务，不存在每天的使用上限，但是由于速度原因存在一个隐含的上限：我们每天只能够对大约8万条地址进行地理编码，因此为总共40多万条数据进行地理编码花费了5天多的时间。该免费服务的缺点是地理编码的质量不是很好（它只使用了公开可以访问的地址数据），但是该服务的创始人很乐于帮助别人，并且发布了该服务主题的优秀的介绍文档(Gldberg 2008）。

除了经度和纬度，USC WebGIS服务提供的结果中还包含了用来描述结果本身准确程度的一个类别变量：精确地址、邮政编码、区县等。

数据检查

对于每个阶段的数据分析，通常而言，花一些时间来确保数据的准确性是完全必要的，地理编码也不例外。地理编码的错误来自于很多方面：地址中存在拓扑错误，新的建筑通常不会列在公开的数据库中，邮政编码随着时间可能变更。我们在使用USC WebGIS过程中，怀疑该软件可能存在一个bug，因为大量的地址被错误地分配到该州的洛杉矶及其他地区；我们使用另一个免费在线服务http：//gpsvisualizer.com对这些地址进行了重映射。调试过程中，为了识别在旧金山海湾区很远以外的地址，我们使用统计建模工具R来为每个区县以及绝大多数的城镇描绘经度和纬度的简单地图。

San Jose地区的地址提出了一个有趣的地理编码挑战。如一些“城镇”的销售列表，我们找到的所有映射网站都无法识别这些信息，因此我们假定它们是社区的非正式名称：San Jose的北部、南部、东部和西部，Berryessa、Cambrian和一些其他地区。

只要可能，我们就试着纠正任何错误。当无法纠正错误时，我们就使用统计建模工具R的缺失值来表示我们不知道该地区准确的经纬度值。这种处理方式比扔到那些“错误匹配”更好，因为不同目的需要有不同的准确度：当我们以区县或者城市的粒度对数据进行映射时，只要有很接近的地理位置就可以。为经度和纬度使用缺失值确保任何包含可疑的地理编码的地区都会从使用经纬度的分析中删除，但是在其他情况

中，都会包含这些缺失值。

分析

为了对房地产市场变化有更广泛的概览，我们从平均销售价格和销售量谈起。因为数据是以周为单位生成的，这是一个自然的时间单位。

图18-1显示了293周数据中每周的平均销售价格和销售量。这里存在一些非常有趣的模式。平均价格的行为非常显著：一直呈上升趋势直到2007年7月，然后是急剧下降，直到现在——房地产市场价格的繁荣和萧条的一个明显的例证。

销售情况则完全不同。大多数年份（尤其是2004年和2005年）显示非常明显的季节效应，夏季中下旬达到高峰，而冬季的销售量则减少。（在这里需要做一个说明，我们使用的这份数据中只有很少一部分提供了真正的截止日期，因此我们使用了在报纸中报道的销售日期，这个日期可能在截止日期的4～6周之后。）一旦我们考虑过去的季节性影响，我们可以看到一些其他方面。从2006年中～2008年初，销售量下降，当然意味着房地产的萧条。但是，在2008年初的销售急剧下降，也预示了冬季销售量的下跌。2008年初的销售增加是怎么回事呢？一个可能是那时房价已经下降了很多，买家在春天到来前购买降价商品。销售量增加的另一种可能是由于有些房子丧失了抵押品赎回权。可能在未来几个月之后，对于给定问题的解释会逐步明朗起来。

这些简单的绘图为进一步探索提出了一些方向。这些模式在所有价格区间对于所有家庭都一样吗？那不同的城市或者是一个城市的社区内

会是什么情况呢？为了调查这些问题，我们将遵循相同的过程：我们将通过不同的方式对数据进行分区，比较不同分区的模式。我们将基于房屋价格和物理位置创建分区（从最贵的到最便宜的），都是城市之间和一个城市内的（旧金山）。

通货膨胀的影响

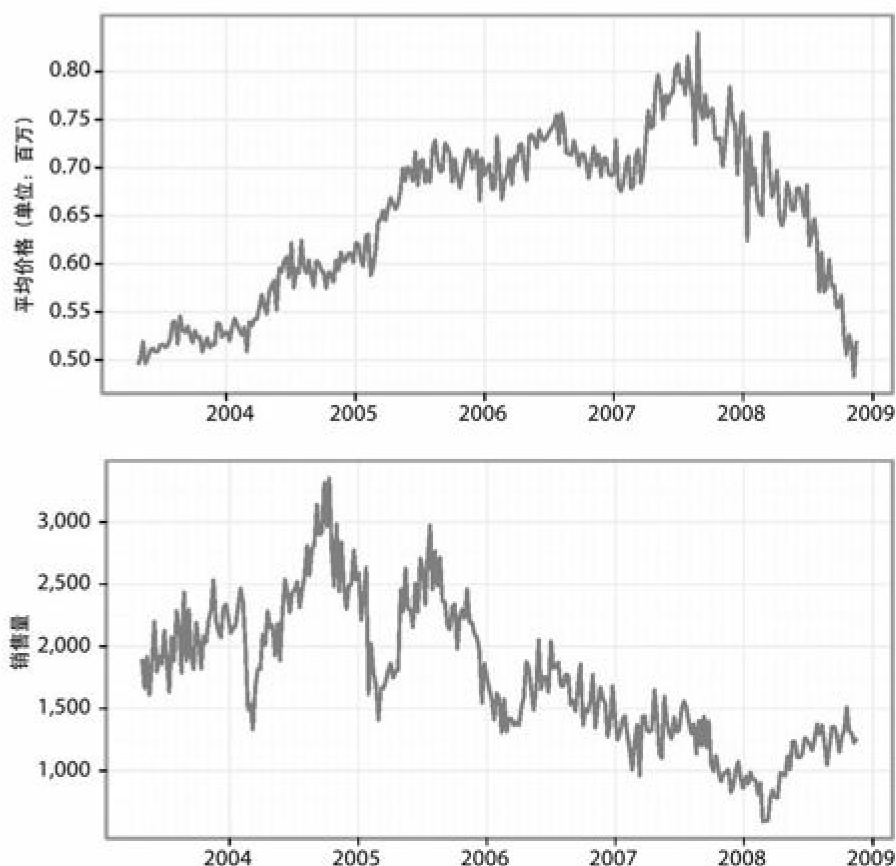


图 18-1：每周的平均价格（上图）和销售量（下图），住房市场繁荣和萧条的明显证据。注意在2008年的销售上升

在进一步分析之前，我们停下来考虑一下通货膨胀。数据是在相对较短的时间周期（大约6年）内收集的，但是我们在思考是否要根据通货膨胀对价格指数做出调整，以便确保2003年支付的价格和2008年支付的价格具有可比性。常用的计算通货膨胀方法是参考消费者价格指数(Cnsumer Price Index,CPI)，它是由劳工统计局生成的，可以访问

http：//www.bls.gov/CPI获取。CPI计算经常购买的消费者产品和服务的加权的“桶”的价格。该价格是每个月计算一次，我们将使用西海岸序列(CUR0400SA0序列）为通货膨胀对数据做调整。我们想把所有的值都调整为2003年的价格，因此，我们把每个CPI值除以其在2003年3月相应的价格值。这种操作被称为“相关价格指数化”(Idexing)。它得到的是2003年1美元在每个时间点的相对价值，使得可以很容易看懂图中的通货膨胀效果：值1.1表示从数据起始开始，累积通货膨胀达到10%。图18-2显示基于CPI的通货膨胀衡量尺度以及为通货膨胀调整价格的影响。通货膨胀在过去5年一直稳定增长，我们可以发现根据通货膨胀调整的房价的增长趋势稍微低于未调整的趋势。最后，我们还是决定不按照通货膨胀调整销售价格。住房价格对于CPI有影响，因为它的一个子指数是住房指数，它是出租和“拥有者的等价出租”的尺度。可能有人会说住房价格对于CPI在整个研究期间内有深远的影响。

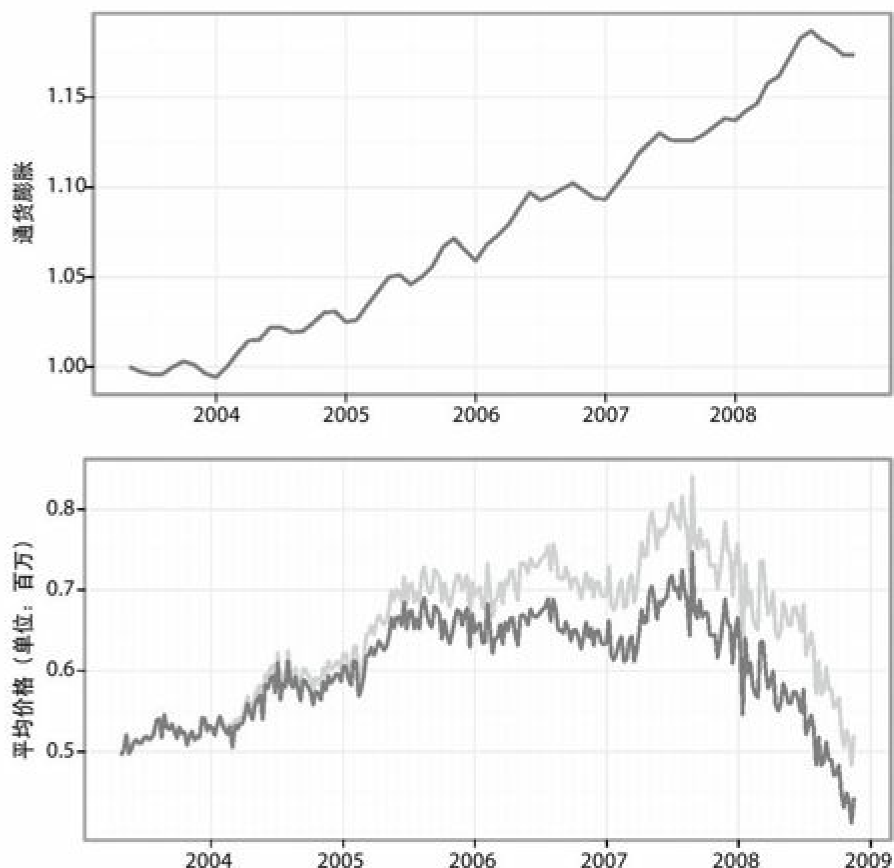


图　18-2：（上图）通货膨胀，在序列开始时价格相关指数化为1；（下图）以2003年价格为标准根据通货膨胀调整的住房价格（红色）和未调整的价格（蓝色）。没有根据通货膨胀调整价格使得价格上升趋势看起来更陡峭，但是对于价格下降几乎没有任何影响。不包含Monterey、San Benito、San Joaquin和Santa Cruz区县，因为我们只有这些区县的2008年的数据

有了这些基本概况，我们现在开始深入到细节。在本章的后面部分，我们把住房销售划分成几个小的群体，首先是通过价格，然后是通过地理位置。我们对于发现住房市场是否对某些业主群体比其他人带来更多的影响感兴趣。

富者更富，穷者更穷

住房危机给富人带来的影响和穷人带来的影响是相同的吗？该危机的影响是否改善或者恶化这两个群体的相对平等性？在本章中，我们将探索危机如何影响住房价格的分布。一个重要的提醒是我们查看的是洛杉矶海湾区，因此，其住房价格会比该区的其他地方的价格高得多，但是我们仍然期望能够观察到一些相对的不平等性。（注意：在以下部分，我们将频繁地使用单词“住房”(huses)来泛指住宅房地产的所有领域：住房、别墅、公寓等。）

我们先计算每个月的价格十分位数。该十分位数分别是住房价格下降10%、20%、30%、40%、50%、60%、70%、80%和90%。这是对每月价格分布的简明摘要：不是像我们之前所做的那样查看平均价格，而是通过9个数值来总结价格的完整分布。（我们没有显示最低价格或者最高价格的曲线，因为这些价格太不稳定。）

图18-3显示了这些十分位数如何随着时间变化。最高行是第九个十等分，价格低于住房价格的90%；最低行是第一个十等分，价格仅仅比住房价格低10%。中间行是平均值，该价格是住房价格的一半。线条颜色从深变浅，表示从最贵的到最便宜的。每条线有相似的模式，我们可以观察到受2007年的住房泡沫的影响，尤其是价格最高的住房。

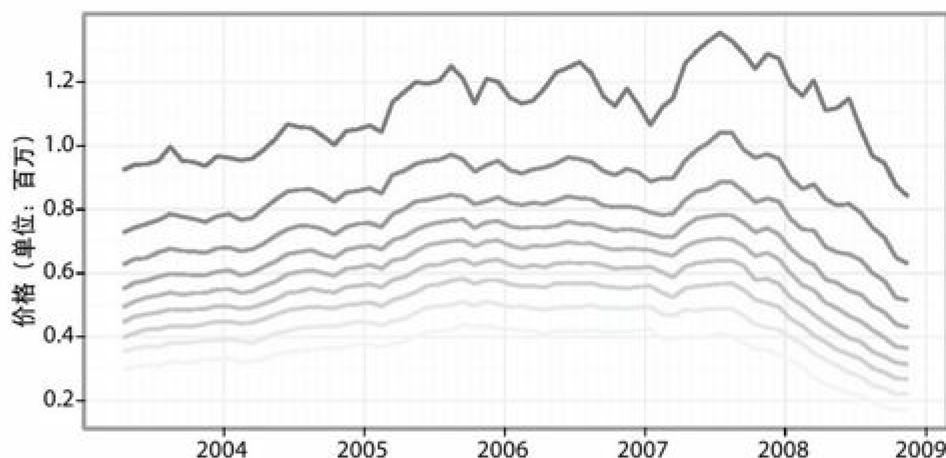


图　18-3：每个百分点内的每月平均住房价格。百分点越低，颜色越浅。该绘图很清晰地显示了价格更高的住宅的泡沫的本质，但是对于最低价格区间，它不是很明显

该绘图允许我们比较每个十分位数的绝对值，但是查看相对价格可能更合适：价格如何相应的变化？查看相对价格的一种方式是比较每个十分位数和它的初始值。为了实现这一点，我们对每个十分位数指数化，通过其初始值划分序列，正如我们计算消费价格指数一样。图18-4显示了这些指数。每个十分数起始位1.0，我们可以看到随时间的价格相对变化。该图的一个很有趣的一面是价格更低的住房（浅色线条）所达到的峰值更高更早（2005年中），然后在后期下降更快。（注意深色和浅色线条在2007年早期的位置交替。）最便宜的住房，即在最低的十分数，损失值相当于2003年价值的43%，而最昂贵的住房只损失了9%。比较图18-3和图18-4，我们发现虽然在实际价格中，最大的绝对下降值发生在最贵的住房，但是，最便宜的住房相对损失值是最大的。

图18-5是查看该不平等性的另一种方式。我们把所有的价格都除以

住房平均价格。其值表示平均住房价格：值1.2表示价格高出平均价格20%，0.8表示低于平均价格20%。自从2007年年初，虽然价格高的住房形式还很繁荣，相对不平等性已经开始增长。这是否意味着住房高的和住房低的价格差距更大是后继产生危机的前兆？该情况是否暗示了其他危机？这些问题可以进一步研究，但是我们这里没有数据来对它们进行深入探索。

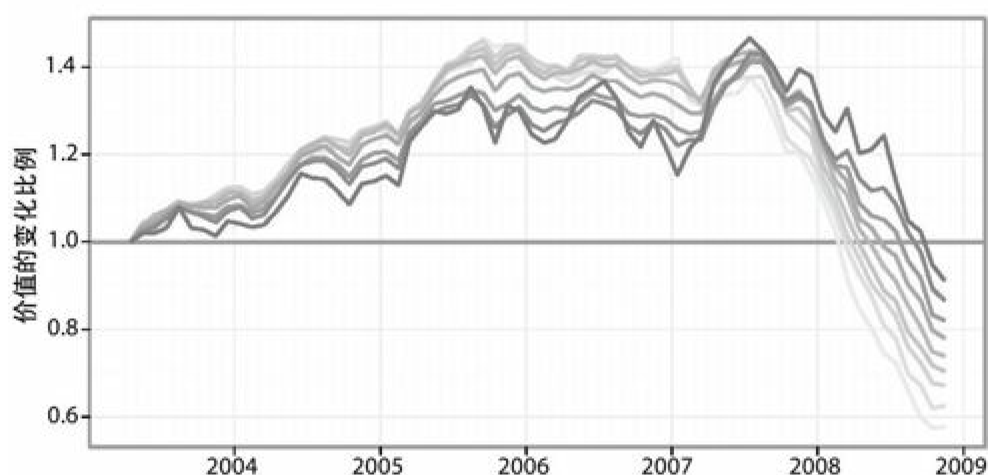


图　18-4：每个十分数内指数化的住房价格（线条颜色越浅，价格越低）萧条期在价格低的住房中首先发生：价值越低的住房的平均价格上升的峰值越高而且越早，下降更快且幅度更大

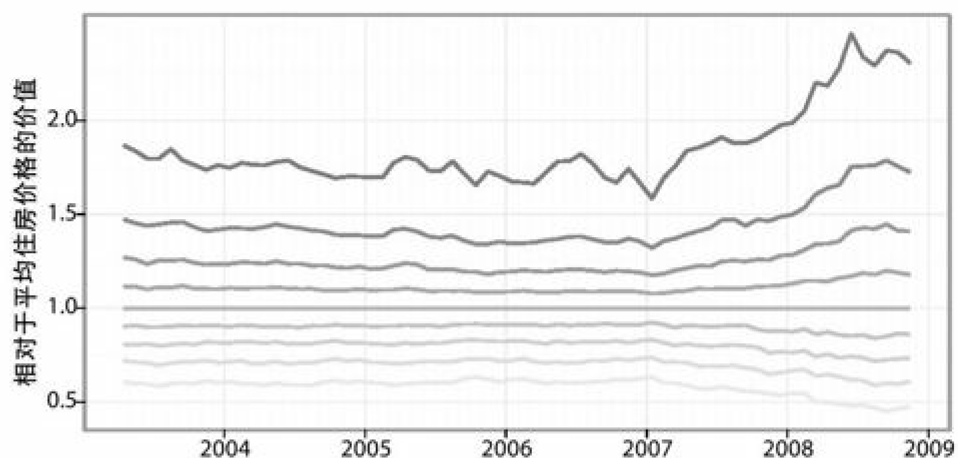


图　18-5：住房价格相对于中等价格的住房的价格。住房价格上存在的差距从2007年早期开始一直增长

地理区别

本节中我们将探索海湾地区不同城市的住房价格变化。因为我们查看的是平均价格，我们必须注意不要包含只有很少的销售量的城市。我们决定重点放在每周平均销售量至少10套的所有城市。这种方式最后筛选出58个城市（总共是245个城市，占24%），共428415套住房销售量（占销售总量的82%）。

然后，我们计算了以周为单位的平均房价。图18-6显示了这些价格，每个城市使用一条不同的曲线来描绘。统计学家为这种展示方式起了一个很生动的名字："意大利面条曲线"。很难从一大堆线条中查看任何东西。对这种曲线的一种改进方法是对每条线进行平滑，删除短期的变化，使我们可以重点观察长期趋势。

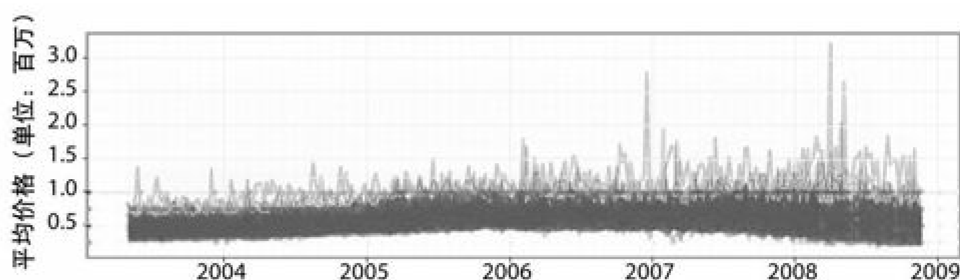


图 18-6：每个城市的每周平均销售价格。这种类型的绘图通常被称为"意大利面条曲线"。它需要平滑，因为曲线中这种"周到周"之间的变化无法检测出趋势

为了创建平滑的曲线，我们使用了广义的可加模型(GM)，它是对线性模型的范化(Wod 2006）。广义可加模型通过优化权衡切近数据并且

线条平滑的方式来拟合线条曲线。实际上，它消除了噪音数据带来的短期影响，并且将重点集中于长期趋势。这正是我们所需要的：我们对每天的或者每周的变化不感兴趣，而只对和住房危机相关的长期变化感兴趣。

图18-7上方显示了平滑化后的结果。这是一个很大的改善。现在我们可以真正看到一些模式！注意该图和第一个图在规模上的重大区别：平滑数据已经删除了一些非常昂贵的住房的销售高峰的描述。我们还会对每个城市指数化，正如我们之前对每个十分数指数化那样：通过相对初始价格进行划分，使得每个城市可以有一个共同的规模，允许我们把重点放在变化上。这一点显示在图18-7下方。

现在还存在很多变化，但是我们可以开始看到增长值模式直到2007年年中，然后后期值开始下降。为了进一步探讨，我们需要分别查看这些城市，如图18-8所示。该曲线图占了很大空间，但是它提供了很多额外的信息，因此是值得的。我们可以挑选出一些有趣的模式：伯克利和旧金山显示的高峰值较低，而下降值较少；而Mountain View的独特之处在于它的住房价格一点都没有下降。其他城市如Oakley、Vallejo和San Pablo，则是大起大落，高峰值很高，而跌落值也很大。

回想一下我们之前关于San Jose城市的讨论，我们注意到，原始数据本身描述了San Jose城市的很多社区。由于这一点，有时相同的地址被分配给多个社区，但是该数据还表明这些社区具有鲜明的特点。Berryessa、东San Jose、北San Jose和南San Jose有相似的曲线，呈现出

很鲜明的高峰和一样鲜明的下降；而另一方面，Cambrian、San Jose和西San Jose则没有出现这么大幅度的下降。

经过进一步调查，我们发现一个使得城市之间相互区别的主要特征：在繁荣期价格之间的区别以及它们最近的下降幅度。我们创建了一个新的变量，称为“价格回落”，它表示在2006年2月（繁荣期的顶峰）和2008年11月（写本章内容时的“低谷”）之间的平均价格的相对下降。图18-9通过新的变量对城市进行分组。这些城市的划分是随机的，但是你可以看到每组城市如何遵循相似的模式：繁荣越鼎盛，崩溃越惨烈。这意味着这单个数字可以很好地总结住房危机的繁荣和萧条方面。

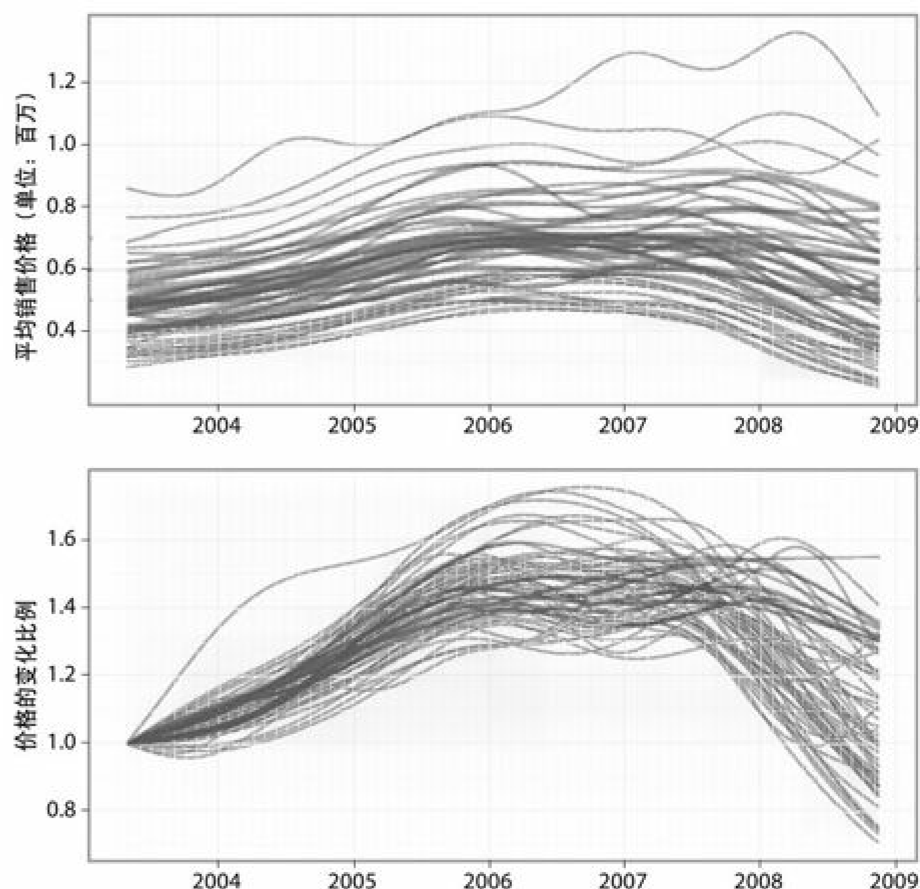


图 18-7：平滑后的每周销售价格，每条曲线表示一个城市（上

图）。下图底部的曲线进行了指数化来显示价格的变化比例。模式开始出现了

我们已经确定城市具有不同的模式，但是还不知道产生这种现象的可能原因。地理模式，如图18-10所示，没有显示任何特别突出的，除了受灾最严重的地区通常是旧金山市的北部和东部。这并不能提供有力的说服力，因此我们需要寻找更多的数据，可以帮助我们获得更深入的理解。

人口普查信息

美国人口普查局提供了在县、市两个级别的最新调查的人口统计信息。quickfacts网站（比如http：//quickfacts.census.gov/qfd/states/06/0649670.html)显示了每个城市的很多有趣的人口统计变量。遗憾的是，城市级别的数据无法简单地通过下载获取，但是我们可以采用脚本（正如那些我们使用的销售数据）来收集人口统计信息，并把它转化为csv格式。此外，在人口普查数据和销售数据中，城市的定义有一点区别，因此我们从全部58个城市中只匹配了46个。人口统计数据没有覆盖我们选择的一些城市，因为这些城市的人口低于某些截断值（阈值）；而另一方面正如我们之前讨论San Jose城市时所指出的，一些住房数据称之为“城市”的，实际上是更大城市范围内的社区。

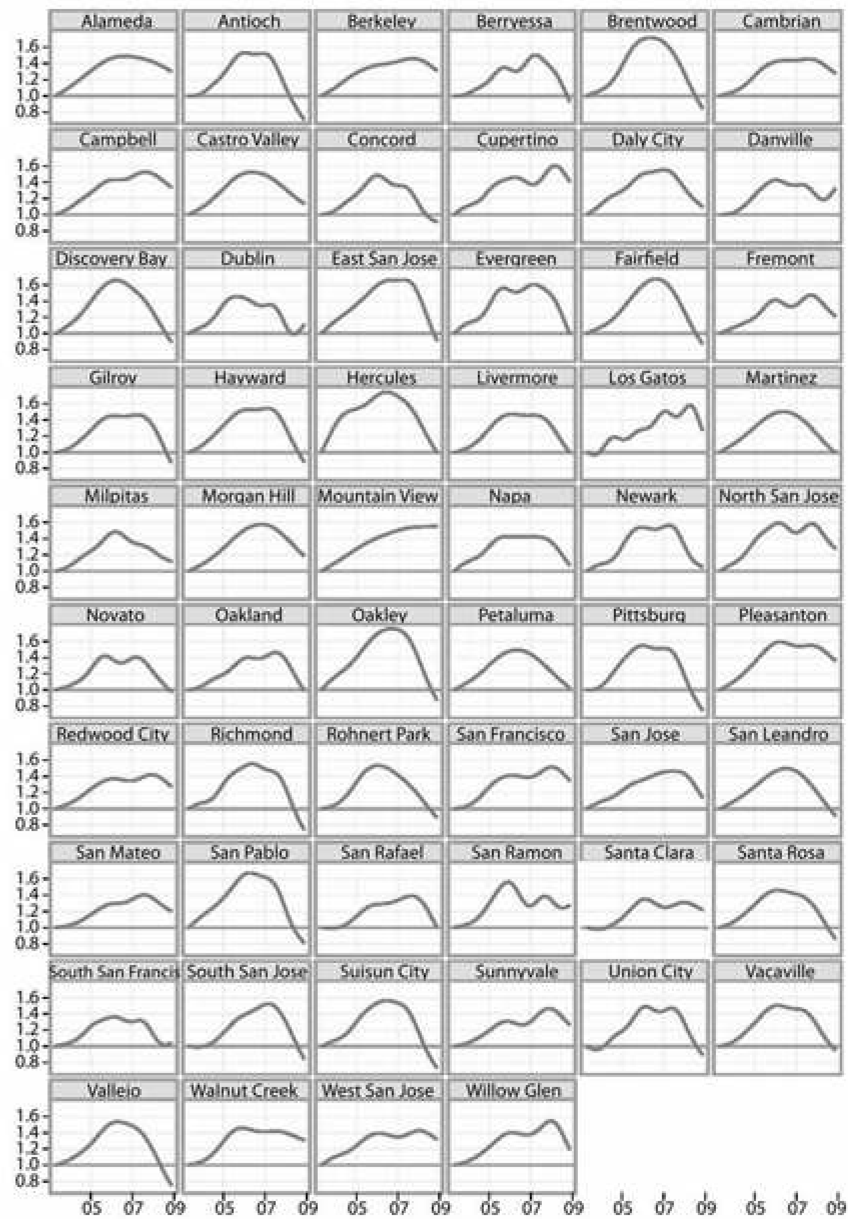


图　18-8：每个城市的销售价格图，已经进行了平滑化和指数化。这些曲线和图18-7中下方的图的各条交叠的曲线（意大利面条曲线）完全相同。San Pablo的曲线显示住房市场的繁荣和萧条；伯克利的曲线显示变化幅度更小；Mountain View则是唯一一个价格还在上升的城市

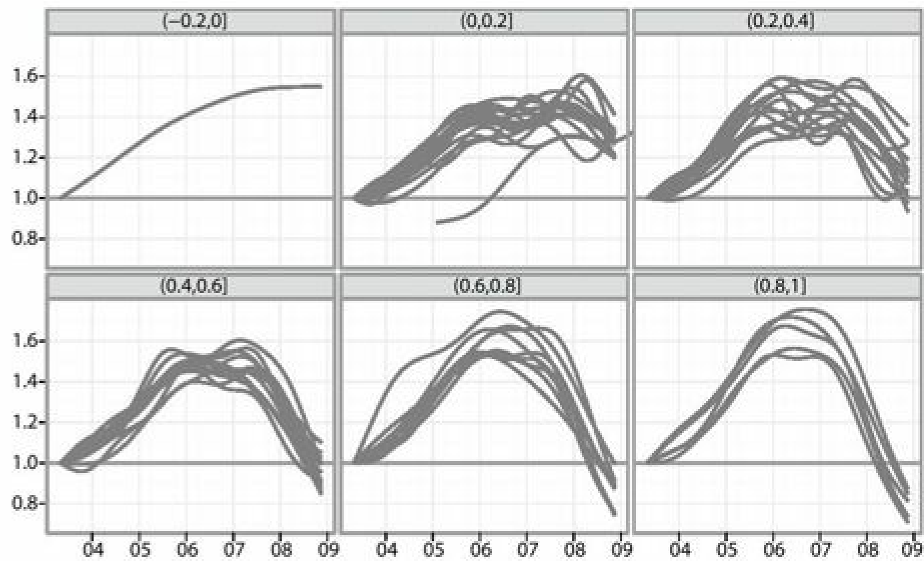


图　18-9：根据城镇的价格回落对曲线进行分组图。图中左上方的城镇价格下降最大（在0.8和1之间，或80%打破100%）；图中右下方的城镇(Muntain View)是唯一价格没有下降的。每个组内的模式是相似的，表明单个数字提供了很有用的方式把城市划分成组

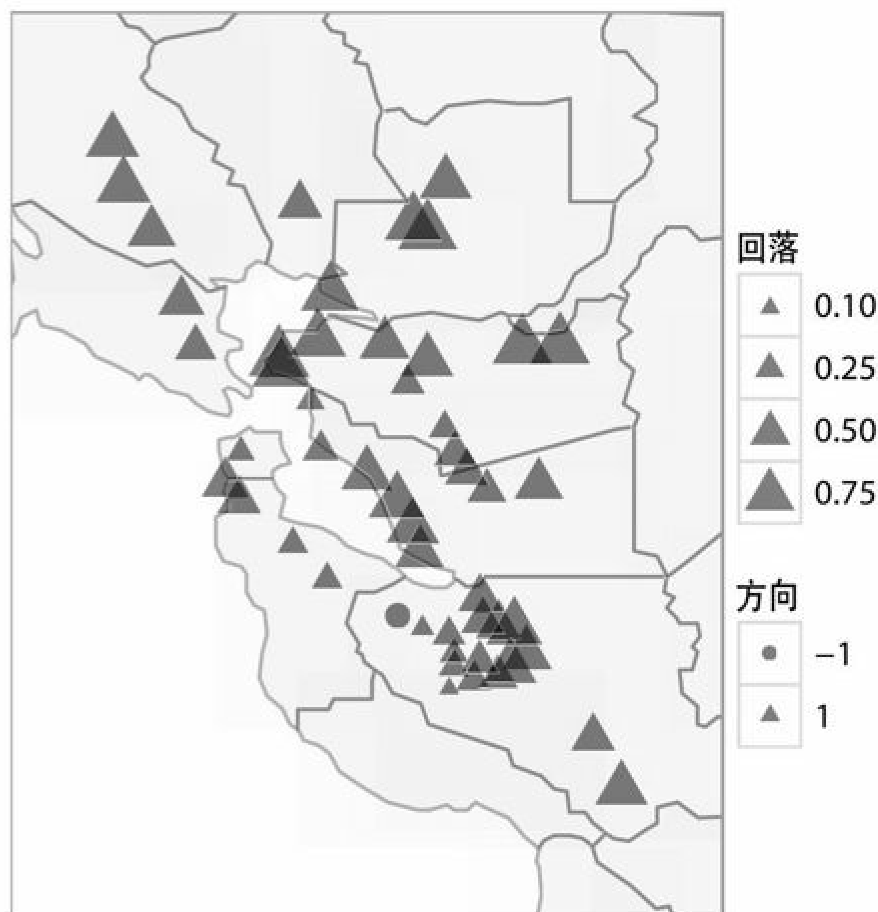


图　18-10：价格回落的地理分布。降价最严重的城镇是北部和东部。单个环表示Mountain View的地理位置，它是唯一销售价格继续增长的城市

一看人口统计变量就能发现影响最严重的城市有大量的婴儿和小孩，家庭成员更多，学士学位更少，交通时间更长。更重要的是，这些城市的平均收入也更低，这可能是导致其他关系的主要因素。图18-11包含了三个散点图，说明了住房价格回落和收入、大学生比例以及交通时间的关系。价格回落和交通之间的关联度很低，但是注意交通时间最长（超过35分钟）的所有城市在价格上有特别大的回落。住房危机看起

来在更贫困的地区，受到的冲击影响更大。

县级人口普查数据包含比城市数据更多的变量，因此我们分析了县级数据来进一步解释住房危机。图18-12中上方的图显示，对于我们有销售数据的任何县，描绘了住房单元个数的变化比例（从2000～2006年）和2008年的平均销售价格比较。在最近住房价值和新建筑数据之间存在很强的负面关系。换句话说，最近几年绝大多数的建筑繁荣发生在更贫困的社区，正如我们之前所述，有些地区其后继的价格疲软是最严重的。特别是San Joaquin县，它的城镇价格都很低，从最近几年到现在为止，有更多的新建筑。应该注意的是，这些城市中，有些我们没有很多的销售数据（比如San Benito和Santa Cruz)，但是这些关系的全局的本质还是非常清晰的。图18-12中下方的图更进一步说明了这个效果，它也显示了住房单元从2000～2006年的比例变化，但是它是和县级的2005年人均收入比较，从人口普查数据中获得。我们注意到该图和前一个图存在相似性，而且它进一步说明了新建筑的密度在相对不富裕的地区更大，即使将统计的聚集级别从城市扩大到县级时，仍然具有上述特征。

显然，价格和收入是强正相关，从图18-11的城市等级上可以观察到。

根据《New York Times》的一篇文章(MKinley 2007），Stockton城市，San Joaquin省最大的城市之一，到2007年夏天，已经是美国丧失抵押品赎回权比例最高的城市了。

遗憾的是，我们没有Stockton城市在2008年之前的销售数据，但是直

到2008年，看起来它仍然是该地区持续低迷的主要代表。Stockton城市的人口在过去10年增长迅速，为了逃避海湾地区过热的房地产市场，上下班的人们住的地方越来越偏远。这有助于解释之前提出的新建筑，而且也把我们的观察和上下班时间关联起来。该文章还列举了Modesto和Merced城市，在中央山谷(Cntral Valley)的两个城镇，在当时都位列全国丧失抵押品赎回权数量最高的十大城市之中。

探索旧金山

探索了城市之间的区别，我们开始更详细地查看单个城市。旧金山是个很显然的选择：在我们包含的数据中它是最大的城市，也是我们最熟悉的城市，而且它包含一些标志性的特征，可以很容易为别人所识别。我们通过抽取旧金山的所有地址来开始探索，这些地址通过非常高的精确度进行编码，为我们总共提供25377个地址。我们创建了一个简单的经纬度散点图，如图18-13所示。

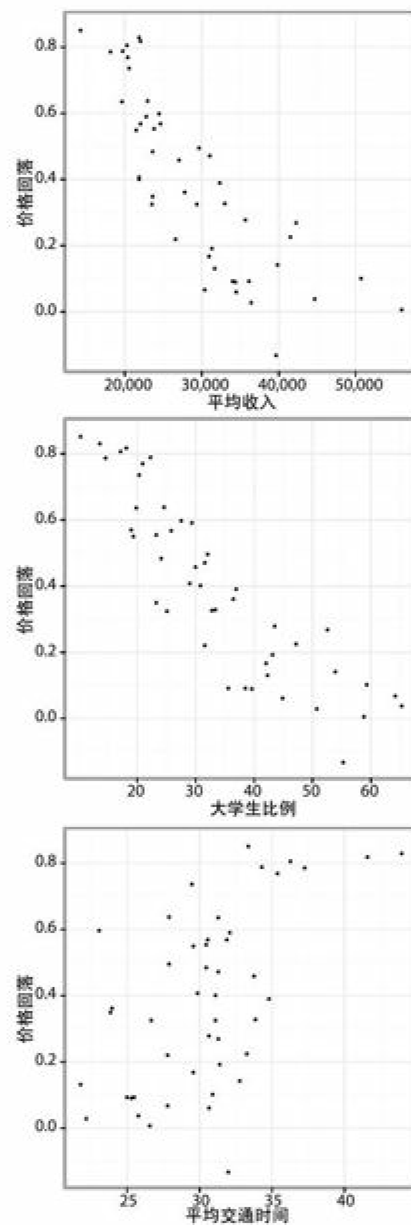


图　18-11：从上到下，住房价格的下降（价格回落）和平均收入、大学生比例和平均交通时间的关系

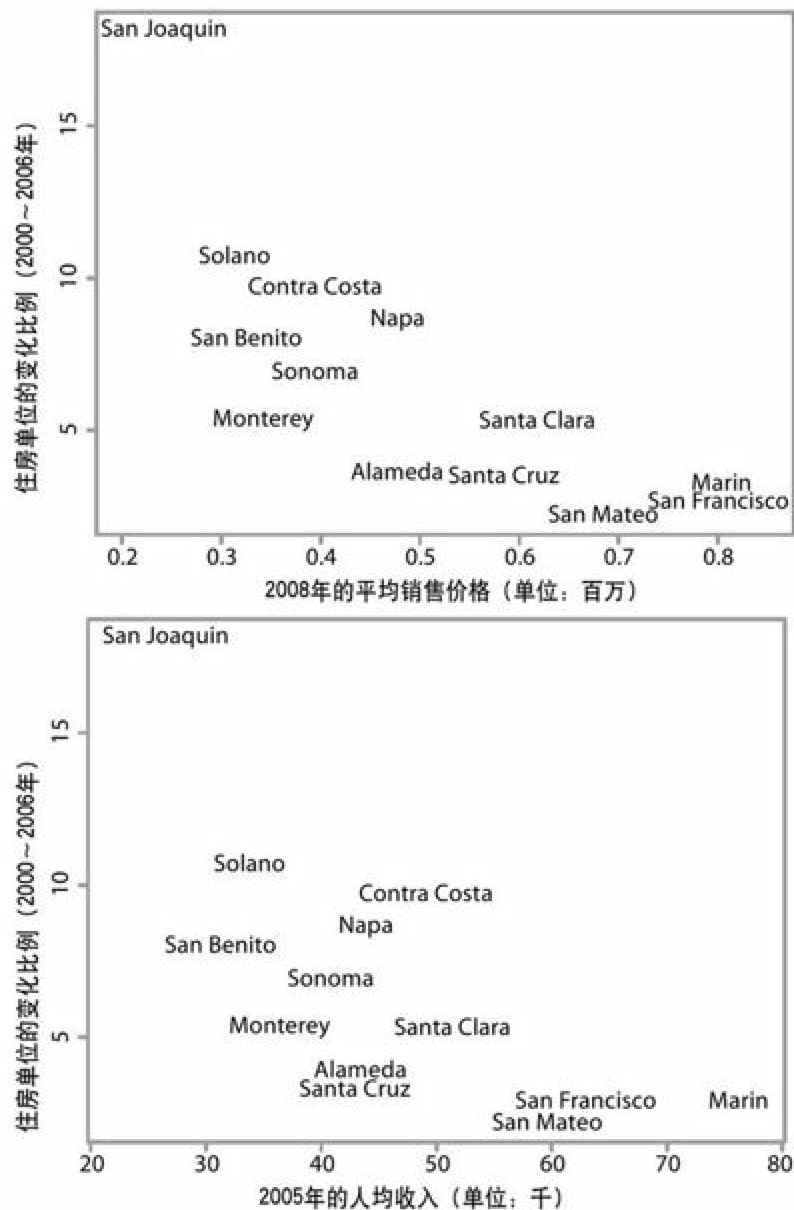


图 18-12：新的构建与最近价格（上图）和个人收入（下图）的关系，数据以县为单位聚集。住房单位个数在价格较低和人均收入较低的住房城市相对增长最大

对于城市的居住地区，该图给了我们非常详尽的画面。我们可以看到街道方向、海滨边界和公园。一些地区的视图，如市中心，显得更散

落，因为这些地方住宅区较少。（在本章中，我们将避免用简写词“房子”，因为很显然很多销售的住房指的是公寓。）

该绘图的一个问题是我们无法看到每个特定位置的销售数量。图18-14显示了试着捕获信息的两种尝试。在上图，我们显示的是冒泡图，其地理位置和销售数量成正比。现在，我们得到和商业区非常不同的视图：那里有很高的销售量。仔细查看这些数据可以发现这些公寓建筑有几百套的公寓。在下图，我们把旧金山分成经度和纬度0.005平方米的方块，计算每个方块内的住房数量。这给了我们更高层次的视图，显示主要的家庭所在的地区。



图　18-13：（上图）对于数据中的每个住宅销售都画出了一个小点，它给我们非常好的旧金山布局的感觉；（下图）为了比较，显示的是一张旧金山的街道地图，来自http：//openstreetmap.com（见彩图68）

使用相同的分块方式，我们计算了住房价格的均值和变化系数。变化系数是通过除以平均值计算出来的标准差。我们这里使用它是因为

$100000的变化在当住房价格很低时要比住房价格高相对更重要。

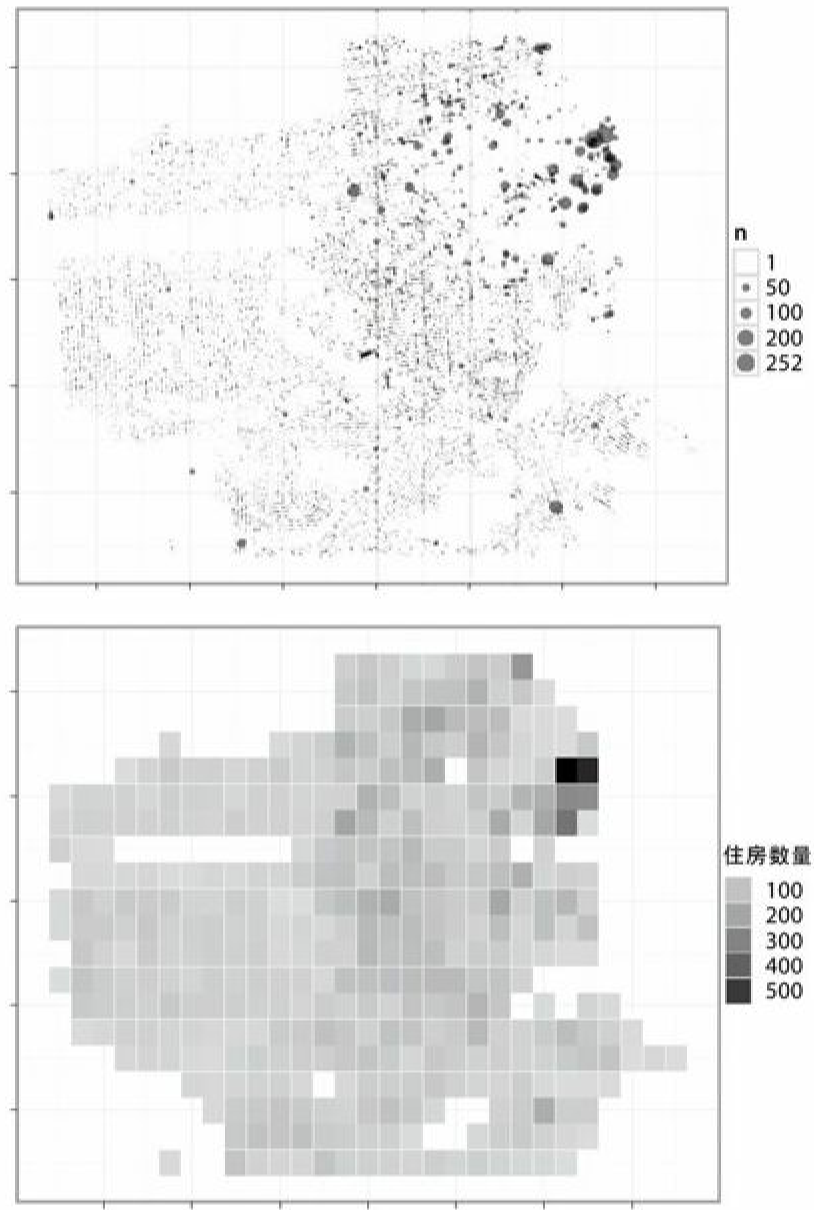


图　18-14：住宅销售数量的地理分布。（上图）该图和之前的绘图相似，但是点的大小和每个唯一位置的销售数量成正比。这大大改变了图片，因为城市大公寓的复杂性现在开始凸显出来。（下图）在更高层聚集的销售显示：经度和纬度被划分到少数的分块，每个分块的销售数

量都被计数，作为分块的颜色显示

图18-15显示了这两种求和统计的地理分布。我们可以看到Presidio城市边界和该城市的南海岸最昂贵的住房。在西南地区看起来有个高峰：这就是富裕的St.Francis Wood地区，在旧金山州立大学附近。变化系数存在一个有趣的地理趋势：似乎价格朝着西北地区增长。

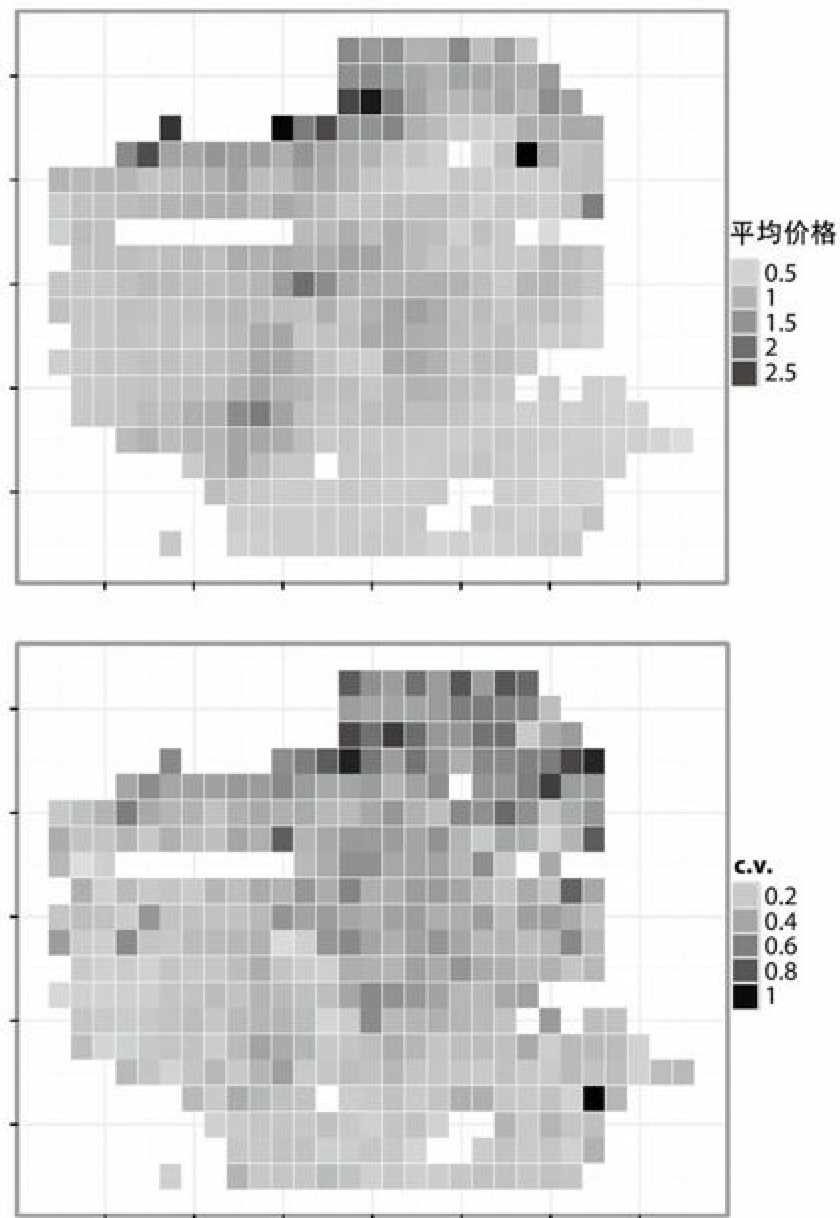


图　18-15：使用和之前图相同的经度和纬度分块方法；均值（上图）和变化系数（下图）是使用相同的灰色调来计算和显示的

结论

我们从多个角度查看了数据，观察到的是相同的东西：住房危机在更贫困地区相对伤害更深。繁荣和萧条对于价格低的住房打击更早更大；平均收入较低的城市高峰值更高，下降更多，落差也更大。很多繁荣景象和新建筑有关，绝大多数的目标是低端市场。

很多新建成的住宅远离旧金山市区，在较不发达的地区，居民平均收入较低、孩子更多、上下班时间更长。虽然价格上最大的绝对下降发生在高端，但从跌下的价格相对于房子本身的价值所占比例的角度考虑，较便宜的住房损失更大。

自从房地产泡沫成为头条新闻，所有这一切都和我们所学到的关于次级抵押贷款一致。很多信誉很低的人也获得了抵押贷款，初始每个月支付很少。当这些支付款项不断增长时，他们就无法支付了，违约和丧失抵押品赎回权的数量开始上升。我们之前猜测过2008年销售的增长可能和丧失抵押品赎回权有关，而且下一步有趣的步骤是定位出丧失抵押品赎回权的数据，并结合我们的销售数据一起调研。

我们使用了相对简单的统计方法如索引、计算位数、平滑和分块来探索大的、复杂的数据集合。我们从广泛的摘要开始，然后深入探索细节，但是我们还只是触及表面。如果你对我们的数据感兴趣并希望进一步了解我们的工作细节，或者尝试一些你的想法，你可以在git库：https：//github.com/hadley/sfhousing检出所有数据和代码。我们写的所有

代码（R、Perl和shell脚本）都是运行在开源软件上，因此，任何人都可以复制这些工作成果，而不需要购买昂贵的软件。再生性的原则(Gntleman和Temple Lang 2007）在科学实验室非常重要，在这里也很重要：如果我们犯了一个错误，你可以发现它、解决它并观察对结论的影响。

通过这次实践工作，我们获得了从住房数据中抽取、探索、分析并最终得出有用的洞察见解的快乐。此外，我们希望关于该策略、方法和技巧的通用的说明能够被证明对于那些和我们志同道合一起工作和从真实数据中学习的人有所帮助。

参考文献

[1]Gentleman,Robert and Duncan Temple Lang. 2007."Statistical analyses and reproducible research."Journal of Computational and Graphical Statistics，16（1）：1-23.

[2]Goldberg,Daniel W. 2008."A geocoding best practices guide."Technical report,GIS Research Laboratory,University of Southern California.http：//www.naaccr.org/filesystem/pdf/Geocoding_Best_Practices.

[3]Goldberg,D. W.， and J.P.Wilson.2008.USC WebGIS Services.https：//webgis.usc.edu.Last accessed December 2008.

[4]McKinley,Jesse."From housing to haven to foreclosure leader."New York Times. August 13，2007.

[5]Wood,Simon. 2006.Generalized Additive Models:An Introduction with R.Boca Raton,FL:Chapman＆Hall/CRC.

第19章　美丽的政治数据

Andrew Gelman、Jonathan P. Kastellec和Yair Ghitza

历史上最早的一些关于数据分析的例子涉及政治和政府；甚至单词“统计”(satistics)也显示了数据收集“为了国家”以及“和国家”之间的关系。一些统计学先驱，包括Playfair、Laplace和Galton，投入了很大努力来设计和分析公共数据。在20世纪，统计学和Gallup投票、经济和军事组织（五年计划等），甚至是像Svengali一样的政治顾问（1964年的小说《The 480》的一个角色，作者还写了《The Ugly American》、《Fail-Safe》以及其他冷战时期的最佳畅销书）。最近，电视观众已经开始习惯于彩色的地图以及最新的按照地区和人口分片的民意调查和选举结果。还有后面更复杂成熟的杂志《USA Today》、《New York Times》和博客网站FiveThirtyEight.com。

本章提供了一些例子，在这些例子中数据可视化可以有助于加深我们对政治的理解，还带有做出每种选择涉及的因素的讨论。在本章中，我们的重点是使用图形进行研究和演示。

我们尝试使用以下模板：

·“图X显示了……”

·“该图中的每个点（或每条线）表示……”

·“这些不同的图形表示……”

·“在做图前，我们做了……但是它效果不好，因为……”

·“一个自然的延伸是……”

我们并没有形成一套完整的统计图理论——最接近于此的尝试是把对图形显示的探索和检查统计模型的拟合度关联起来(Glman 2003）——但是我们希望这一小段内容对读者自己的理解有所帮助。我们认为这些图形不仅仅是漂亮的手工杰作，而是觉得它们是帮助我们领悟真正的美丽的工具。

我们使用工作中的例子来进行说明，不是因为我们的图形特别漂亮，而是因为在这些例子中，我们知道每个图形背后的故事。

实例1：重新划分选区和党派偏好

图19-1显示了重新划分选区对党派偏好产生的影响的估计（重新画线将可以选举出立法议员的区区分开来）。图中的每个点表示一个州议会选举年（如1972年的Missouri)，其纵轴和横轴显示在那次选举和两年前上一次选举中的党派偏好。

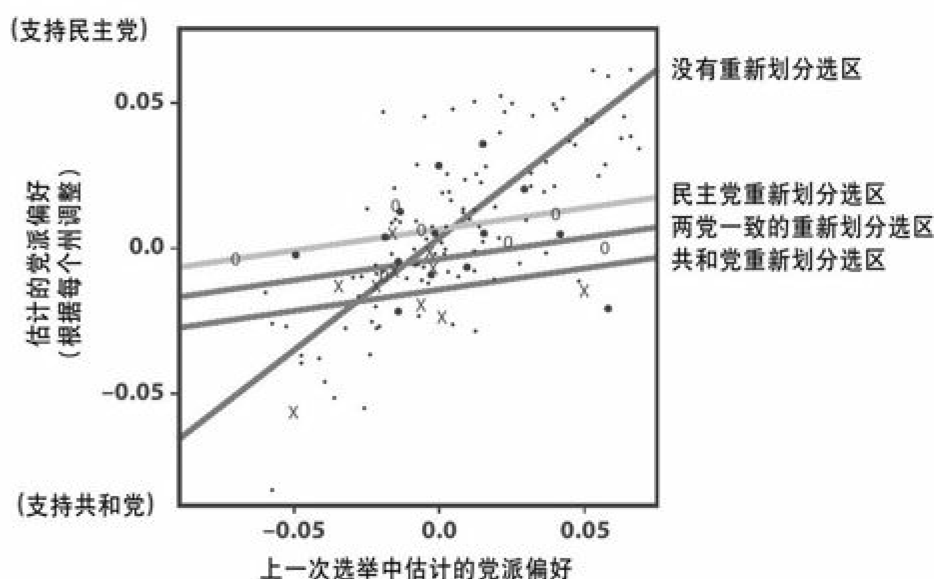


图　19-1：选区重新划分对党派偏好的影响。每个符号表示一个州的选举年，点表示对照组（没有重新划分的年份）以及与不同的重新划分类型一致的其他符号。正如这些拟合线所示，“先验”值比“后验”值的对照组处理（重新划分）情况更具有预测性。处理的显著效果是把党派偏好的期望值置为0，而通过普通的方法无法发现这种效果，普通的方法是拟合模型，假定处理的和对照案例存在并发的回归模型。该图只能显示数据中的关键模式

“党派偏好”，正如这里所定义的，是衡量考虑了投票份额后，选举系统在多大程度上更有利于民主党或共和党。简单地说，党派偏好是民主党在议院立法中期望赢得的议席份额，如果它们的平均值是50%的选票。偏好通常是在-5%～5%，意味着在一个州立法中赢得一般选票的党派将会赢得45%～55%的议席。

图中的小圆点表示没有重新划分选区的“对照”情况，更大的符号对应不同类型的重新划分，这里我们一律视为“被处理”的情况。选举每两年举行一次，而重新划分选区通常每10年发生一次，因此绝大多数的数据点是对照情况。对照情况在衡量前后的关联要远远大于被处理的情况。两组中的斜线区别应该是不足为怪。在没有重新划分选区的对照情况下，州立法几乎没有变化，因此党派偏好可能和之前的选举没有太多变化。相反地，当重新描绘立法区时，将会出现更大更不可预测的变化。为了观察这种效果，在处理中对变量进行建模是非常重要的。

对于民主党从重新划分中获取党派偏好的最简单的方法是划出“区”线，这样在他们所在的每个区都赢得60%的选举权，和共和党的加起来，他们就赢得了接近100%的选举权。然而，这种操纵方式（“不公正的选区划分”）可能在实际情况中是不可行的，即使限制所有区包含相同的人口和区之间的毗连，在美国法院诉讼中，这种潜在的恶意“不公正的选区划分”可能会被推翻。

图19-1是美丽的，因为在我们画出这张图之前(Glman和King 1994），党派内对于重新划分选区的讨论集中于本党是否可以获得较大

的利益，以及是否会削弱选举制度的竞争性（因为描绘区域线条的立法院可以试着为自己和同事保留“安全座位”）。

在我们第一次尝试使用该数据来为选区重新划分的后果建模时，我们采用了没有交互的线性回归模型进行拟合，因此这样做就完全错过了最有价值的内容。最终在将数据和拟合回归线都绘出之后，我们才注意到重点并采用了一个更为合适的模型拟合。

我们的图表显示重新划分的主要结果是降低了党派偏好的波动幅度（而且也使得选举系统更好地反映选民，但这是没有显示在这里的另一个图形的主题。）

实例2：估计的时间序列

图19-2说明了经典的逻辑回归（一种预测“是/否”结果的标准统计方法）方法的一个问题，以及如何使用所谓的弱信息化的贝叶斯方法来解决该问题。对于从1952～2000年每次美国总统选举的投票数据，我们为每年的数据拟合了一个分离的逻辑回归模型，预测给定种族、收入和一些其他变量的共和党人的投票选择。

在每个小图中，每个点代表一个逻辑回归系数，垂直线条表示估计的不确定性程度。点序列显示了每种选择的单独的估计，图形中的两行显示了种族和收入的估计系数的时间序列。（为了简单起见，我们没有将其他的系数列在这里。）图中的最左边一列的两个图形使用的是经典的估计，而最右侧的两列则采用的是两种不同的贝叶斯估计（在这个例子中，它们生成的答案基本相同）。

图19-2的估计结果看起来不错，除了1964年是一个完全的分隔点，那一年所有的黑人都支持民主党。因此，种族系数的估计值是负无穷大——也就是说，其推论是那一年黑人选举共和党的概率是0%。1964年共和党确实不受黑人选民支持（那年共和党候选人是Barry Goldwater，他曾经反对《the Civil Rights Act》），因此他们得到的黑人选票肯定为0。这种回归的目的，正如几乎所有的调查分析一样，是为了得出关于普通人群的结论，而不仅仅是调查一个小样本，因此，我们不能仅仅满足于经典的负无穷大的估计结果。（图19-2左栏显示的估计实际上不是

无限的，那是因为用于拟合模型的软件是迭代的，在达到一定程度之后就停止了。）

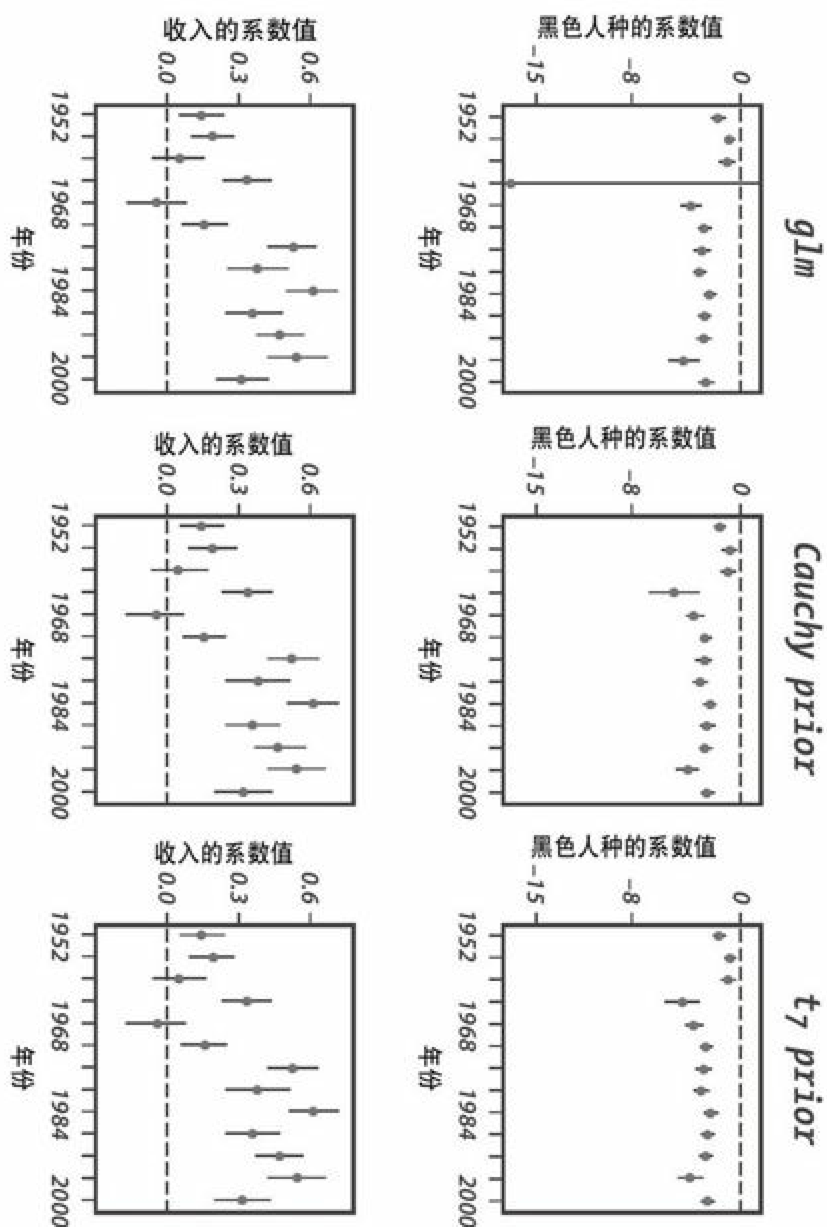


图　19-2：左栏显示了在一个逻辑回归中的两个预测器的估计的系数值（标准误差±1），预测共和党人会选举民主党候选人作为总统的概率，对美国选举研究中心提供的从1952～2000年的每次选举的数据进

行分别拟合。数值变量收入（初始值规模是1-5）首先被集中起来，然后通过除以两种标准差重新调整规模。1964年存在一个完全分割点（没有一个非洲裔的选民支持共和党候选人Barry Goldwater)，导致那一年预测器的系数估计值为-1。（该估计的有限值和标准误差是通过多次使用统计建模工具R中的glm函数进行迭代确定的，直到程序终止。）其他两列显示了对于相同模型，使用不同的“弱信息”先验分布的贝叶斯估计。贝叶斯推论解决了1964年由于完全分割点导致的预测器系数估计值为-1的问题，而对其他年份的估计则没有太多效果

本图以及其他和它类似的图的优点在于，它的严格并行性（1990年Tufte和1967年Bertin提出的“小倍数”思想）允许读者——以及该图的创建者——一次性做出很多比较。

贝叶斯方法，正如图19-2的最右侧两栏所示，为1964年的黑人选民系数生成了一个合理的值——低于1952～2000年中的任何一年，而且有更大的不确定性边界，但不是无限的。在解决该问题时，贝叶斯程序并没有混合其他年份或者模型中的其他变量的系数估计（正如图中第二行的收入系数所示）。

该图谈不上漂亮，但是它说明了一条重要且通用的原则，即图形化不仅仅只是为原始数据做的。统计学中常见的实践是在一张表中显示这种结果，但是好的图形可以通过更少的空间显示更多的信息(Glman等2002）。

从我们的角度考虑，以图形的方式展示参数估计有助于向别人表达

我们的方法的有效性，同时也可以进一步证实我们的估计序列是合理的，而将估计系数放到一个表格中的方式（或者更经典的计算机输出的很长的序列）将无法达到这种效果。

实例3：年龄和选举

紧随奥巴马的历史性选举之后，产生了一个年轻选民在获胜联盟中发挥很大作用的推测。选举后民意测验数据显示奥巴马在年轻人当中特别受欢迎，但是这真的很有新闻价值吗？举个例子，政治顾问Mark Penn在《New York Times》的网站上写道：“显然，绝大多数年轻人都选奥巴马，但是他们也非常支持John Kerry。”Penn的看法是正确的吗？

和通常一样，做出比较的最清晰的方式是使用图形。图19-3显示了结果有四个版本：第一个版本是我们在选举夜做出的基本图形（从CNN网站上得到的选后民意调查数据），然后一个学生在Web上注意到了我们的图形，并发布了一份升级版；然后我们发布了自己的包含了更广时间序列的图形。在每个这种图形中，点和线连接，点表示共和党候选人在最近几次选举中在四个不同年龄组中每个分组所占的两党选票的比例。显然2008年是不同的，因此Mark Penn错了——这是权威人士只看数字不看宏观形势的案例。这就是所有图形的共同优势：一次性显示所有细节和模式。

要了解实际的更为宏观的状况，还需要大量的研究，而且我们并不认为这些揭示了一些简单模式的图形从任何角度可以代替有关随着年龄模式和选举的时间推移的变化模式的更严肃的研究。

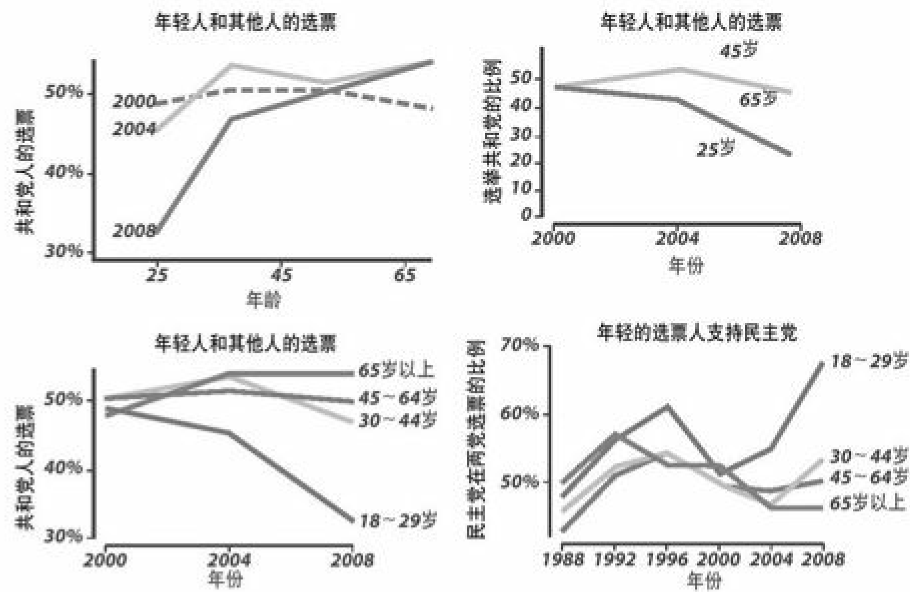


图　19-3：几张显示最近几届美国总统选举的选票年龄纬度的分布模式图

左上图是我们的第一次尝试，基于即时的选后民意调查数据，在选举之夜生成的。右上图是Hober Short创建的，Hober Short是一个学生，他在Web上看到了我们的图，自己做了一个，x轴是选举时间。左下图是基于Short的图的“简洁版”，把所有四个年龄分组直接在图中的线条上标注出来。所有这些图显示了2008年相对于在其之前的两次选举的重大变化。最后，右下图把数据扩展到1988年，显示了1996年Bill Clinton也很受年轻人支持——和奥巴马一样，他也是一个年轻的民主党，对手是年老的共和党——但是没有达到奥巴马在2008年受欢迎的程度。

这些图显示了在制作即使是最简单可行的图形时的选择。正如在很多的政治背景中，最大的收入来自于组合额外的数据——在这种情况下，比较2008年和其之前的年份，比较年轻的选举人和那些年老的选举

人，以及比较三个其他年龄组和另一个（在最后比较中缺乏变化性，成为特别重点研究年轻人的选票趋势的动力）。

此外，我们通过重点研究民主党而不是共和党的选票（由于奥巴马在年轻选民中很受关注，这种方式更合适）以及给图形添加更有描述性的标题来改进图形。

实例4：关于最高法院被提名人的公众舆论和参议院选票

美国参议院做出的决议很少可以像确认或者拒绝最高法院提名人的投票一样向公众公开。但是，很多州选票的结果，比如开支法案或者修改法规，在详细的过程中是不明确或者模糊的，但最高法院提名表决的结果是很明显的：或者确认该提名人，允许她在国家的最高法院任职，或者拒绝该提名人，迫使总统提名另一个候选人(Kstellec等2008）。参议员在投票时是否遵从国家级别的公众意见？

图19-4通过把州级别的公众对九名最高法院法官被提名人的意见和参议员们对他们的投票结果之间的关系放到一张图中的方式初步回答了该问题。在每个图形中，曲线显示了一个参议员投赞成票的概率相对于参议员所在州的公众意见的变化规律。黑色实线是拟合的逻辑回归的估计曲线，而灰色线条集群描绘了这种估计的不确定性。散列标记（或者称“小地毯图片”）表示赞成（“1”）和反对（“0”）提名人的选票，而在每个图的右下角的数字表示该被提名人的总票数。底部的图把所有的被提名人都放到一起。我们通过为每个被提名人增加均值的支持度，使得图形曲线平铺开来，渐次向下。

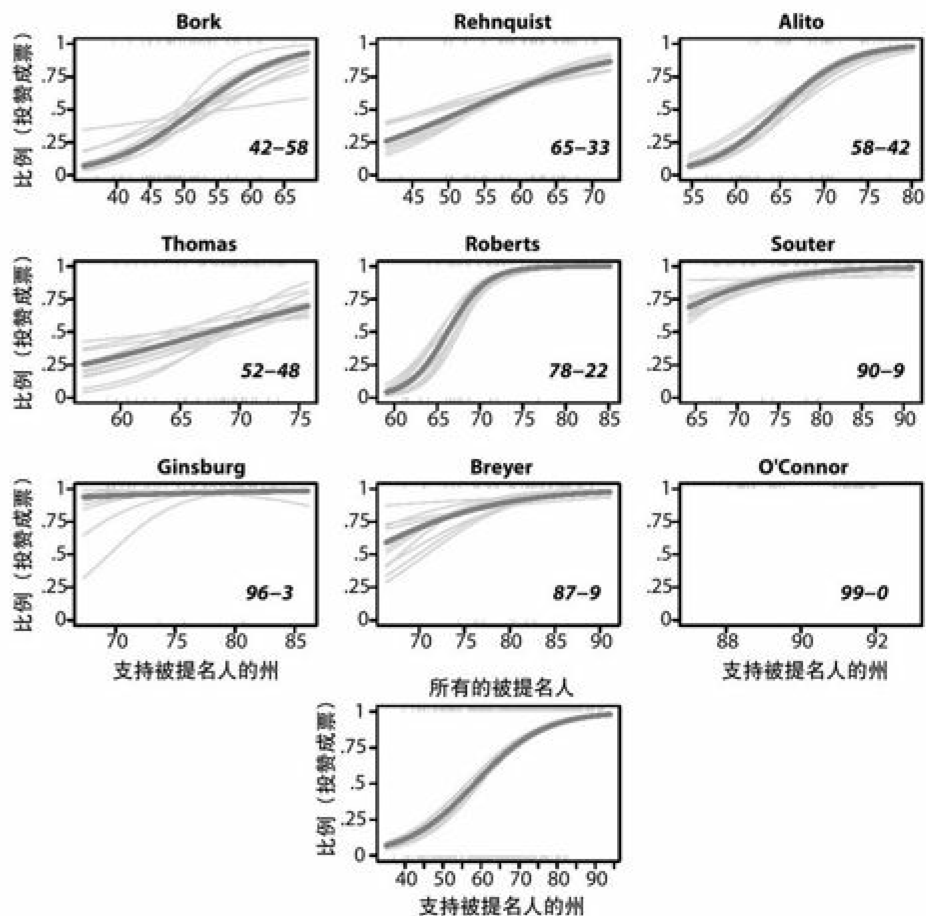


图　19-4：州内公众的意见和该州的参议员在最高法院被提名人投票之间的关联关系。对于每个被提名人，黑色线条描绘的是一个州的参议员投赞成票的概率相对该州的公众舆论的逻辑回归曲线。浅灰色线条描绘了在这些估计中的不确定性。散列标识表示对被提名人的赞成（“1”）和反对（“0”）的选票，而在每个图的右下角的数字表示参议院的总票数。最下面的图把所有的提名人结合在了一起。该图的美丽在于它将原始数据和简单的推理模型融合在了一张图中

该图表明虽然在不同被提名人之间有区别，但公众舆论和投赞成票之间通常成正比关系。毫不奇怪，拥有一边倒的赞成选票的被提名人存

在更大的不确定性。同时，“所有被提名人”的图形显示，通常来说，随着州内公众对提名人的支持的增长，该州的参议员更有可能会投赞成票。（这种关系依然成立，即使一个人控制了唱名投票的其他预测因子，如被提名人的能力以及参议员和被提名人之间的观念差距。）

该图之美在于它把原始数据和简单的推理之美结合起来。通常来说，二元关系是通过表格形式显示的。在这个例子中，这么做需要9个关联系数或者回归系数，以及9个回归模型的标准错误，这会使得表格很丑陋，使它难以对公众舆论和为每个被提名人投票之间的关系进行可视化，而且对于不同被提名人之间的比较也变得困难。我们在图中包含的真正数字（我们通过不显眼的方式，不会转移对图形本身的注意力）是唱名投票差值，它易于理解而且为读者反映出每次提名竞争的激烈程度。最后，如图19-2所示，使用小乘数允许读者马上做出一些比较，避免在一张图中出现信息过载。

实例5：宾夕法尼亚州的本地党派

1986年，政治战略家詹姆斯·卡维尔，后来主持克林顿的首次总统竞选，认为宾夕法尼亚州是Paoli和Penn Hills以及它们之间的Alabama。Paoli是费城的一个郊区，Penn Hills是匹兹堡的一个郊区，因此卡维尔指的是这两个长期“摇摆州”(sing state)的市中心是民主党的堡垒，而该州剩下的农村地区是共和党的支持区。

卡维尔的话意味着公众和最高级别的政治专家广泛存在的把国家分成“红区”和“蓝区”的想法。对于绝大多数熟悉21世纪竞选的美国人来说，关于最近美国政治的最深刻的印象之一是从2000～2004年无处不在的选举地图，以北方和西海岸是“蓝区”州，南方和中心地区为“红区”州为特征。虽然当选为总统的奥巴马坚持我们不是一个“红区”州和“蓝区”州的集合，但是这种根深蒂固的形象很难被克服。

图19-5展示了卡维尔对宾夕法尼亚州的描述以及查看地理党派的不同方式，基于新的和令人兴奋的数据类型以及丰富的可视化技术。地图下侧显示了通过2004年总统大选公布的数据对宾夕法尼亚州的不同地区着色，蓝色表示对民主党候选人John Kerry的支持度更高，红色表示对共和党候选人George W.Bush的支持度更高，紫色表示居于二者之间。通过使用连续的红色-紫色-蓝色模式，而不是更普通的红色实线或蓝色实线来表示每个县的胜出者，我们可以更好地对整个州范围内党派的变化程度进行可视化。

该地图的最上层——分散的圆柱面——显示了在该州随机选取的4000个注册的投票人样例的本地化的党派。本地化党派是衡量民主党或者共和党在每个社区的集中程度。特别地，它是定义为居住在1公里范围内支持民主党的人们的比例。每个圆柱面都是基于在投票人的住宅为中心，以1公里范围内为界限，因此复制该党派衡量方式的区域。蓝色圆柱面表示更支持民主党的地区——这次考虑个人层次的注册——红色圆柱面表示更支持共和党的地区，而紫色表示介于二者之间的区域。

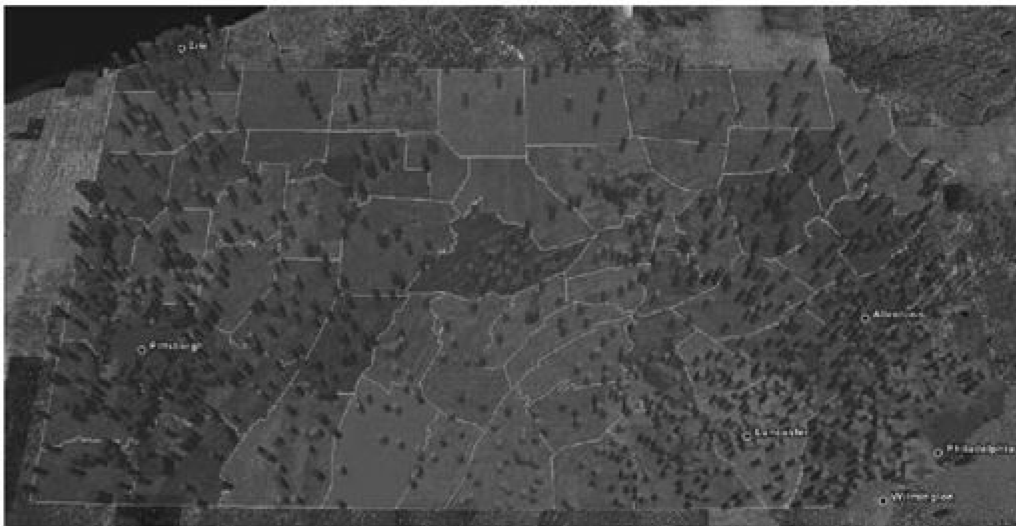


图　19-5：宾夕法尼亚州的地理党派。基础层显示了根据2004年总统竞选公布的数据对宾夕法尼亚州进行的分块着色显示，蓝色表示对民主党候选人John　　Kerry的支持度更高，红色表示对共和党候选人George W.Bush的支持度更高，而紫色分区表示介于这二者之间。分散的柱面图表示对于该州4000个随机注册的投票人的本地党派，定义为居住在1公里范围内支持民主党的人们的概率。每个圆柱面都是基于在投票人的住宅为中心，1公里范围内为界限，因此复制该党派衡量方式的区域。蓝

色圆柱面表示更支持民主党的地区——这次考虑个人层次的注册——红色圆柱面表示更支持共和党的地区，而紫色表示介于二者之间的区域。该图之美在于它显示了在国家级别、州级别，甚至是县级别，红区和蓝区的思想观念的复杂性（见彩图69）

该图之美在于它显示了在国家级别、州级别，甚至是县级别，红区和蓝区的思想观念的复杂性。虽然有时很容易想起红色的州和蓝色的州，但是该图说明了在社区级别（甚至是个人级别）还有紫色。在费城之外，该州最大的城市，你可以发现一些红色的社区。相反地，即使是在该州中部最红的县，仍然存在紫色和蓝色区域。

该图之美还在于它证明了我们通常持有的信念可能受到挑战，以及我们的理解可以通过认真地数据分析和可视化进一步加深。该图使用了Catalist提供的数据，Catalist是一家公司，它维护所有在美国的选举年龄的个人的国家数据库。正如详细的和大规模的数据源变得越来越容易访问，多层可视化技术可以帮助我们使用数据来了解我们周围的世界。

结论

政治数据越来越容易获取，而且在媒体和Web上越来越多地被描绘和分享。在研究领域，政治科学期刊的文章开始利用图形技术来发现和展示结果。从NationMaster.com到Name Voyager(http：//www.babynamewizard.com/voyager)的在线工具变得越来越易于使用，这些工具可以导出数据，如Hans Rosling的TED演讲变得非常受欢迎。我们期望统计可视化在政治分析上变得更重要和更广泛。

参考文献

[1]Bertin,J.（1967）. Semiology of Graphics.Translated by W.J.Berg（1983）.Madison:University of Wisconsin Press.

[2]Gelman,A.（2003）."A Bayesian formulation of exploratory data analysis and goodnessof-fit testing."International Statistical Review 71，369-382.

[3]Gelman,A.， A. Jakulin,M.G.Pittau,and Y.S.Su（2008）."A weakly informative default prior distribution for logistic and other regression models."Annals of Applied Statistics,to appear.

[4]Gelman,A. and G.King（1994）."Enhancing democracy through legislative redistricting."American Political Science Review 88，541-559.

[5]Gelman,A.， C. Pasarica,and R.Dodhia（2002）.“Let's practice what we preach:turning tables into graphs.”American Statistician 56，121-130.

[6]Kastellec,J.， J. Lax,and J.Phillips（2008）.“Public opinion and Senate confirmation of Supreme Court nominees.”Technical report,Department of Political Science,Columbia University.

[7]Tufte,E. R.（1990）.Envisioning Information.Cheshire,CT:Graphics Press.

第20章　连接数据

Toby Segaran

每年，人们都能发明数十种新的或改进的统计和机器学习技术来梳理各种数据集。这些技术几乎无一例外地拥有如下共同点：假定存在一个干净的数据集，该数据集包含现有任务需要的所有信息，而这样的数据集在现实世界中通常是无法找到的。正如亚马逊的前首席科学家Andreas Weigend所言："人们总是问'何种伟大的技术可以应用在这个数据集上'，而实际上他们应该问的是'能够获取到的最好的数据集是什么'"。

同时，科学家们每天通过研究和实验生成TB量级的数据，并把它们放到网上；全世界的各地政府允许下载他们在行政中收集的数据；而用户生成的内容的"繁衍"创造了大量的餐馆数据库、科幻小说以及街道的地理位置，而在此之前根本就没有全面的数据。因此，很多数据是可以获取的，但是除了极少数的专家善于利用这些数据外，其他人很少使用——对于其他所有人来说，不能利用这些数据是很让人遗憾的，如果充分利用，那么一篇文章的一页或者两页，其价值可能会高出10多倍。

我相信当前数据"牧羊人"的最大挑战和机遇在于连接不同的数据集，以便创建新的数据集进行分析，并且充分利用数据的繁衍性，已经研究出来的新技术以及可用的令人不可思议的硬件资源。自从数据库产生之后，数据集成一直是一个问题。但是对于研究人员或者好奇的个人

可以获取的潜在相关的数据量比原来大了好几千倍——该问题已经从企业范围转变为普遍存在的。

对我来说，这是个重大、棘手的问题，它几乎涉及我的职业生涯的方方面面。因此，在本章中，我不会讨论某个项目，而是要采用不同于本书中多数章节所采用的方式，转而探讨我从多年的项目中汲取的经验。

实际上到底存在哪些公共数据

在Freebase(http：//www.freebase.com)中工作时，我研究了数以百计的数据集，这些数据集本身很有趣，但是当作为其他数据集的增强和提供上下文时，则变得更有趣。这些数据集来自非营利机构、政府、公司和基层的努力。以下给出了数据集来源的一个大列表（但是数据样本很小），以飨读者：

责任政治中心(http：//opensecrets.org)公布了美国的政治候选人收到的个人捐款的数量。

·很多国家提供在线的人口普查数据。在美国，你可以从http：//www.census.gov网站上下载人口普查数据。

·Geonames(http：//www. geonames.org)数据库包含了世界各地著名地方的经度、纬度、包含的内容以及级别。

·美国证券交易委员会(http：//sec.gov)提供在美国证券交易所上市的所有公司的可下载的财经数据。

·如环境保护局(http：//epa.gov)这样的机构提供关于某些地方的环境污染以及产生这些污染的设施的可下载的信息。

·一个特别有用的资源是商标数据库(http：//uspto.gov)，可以用它来发现哪些公司拥有哪些品牌，这些品牌名称的卖点是什么以及更重要的和所有不同品牌相关的艺术。

·很多社会网络允许下载很多信息子集，包括关系和其他的如位置之

类的信息。

·几乎和每个可消费产品相关的营养信息（卡路里、脂肪量等）都可以从美国农业部获取(htp：//usda.gov)。

·国家生物技术信息中心(NBI；http：//ncbi.nlm.nih.gov)发布了很多包含了与遗传和医疗信息相关的数据库，包括Genbank、Pubmed、基因和dbSNP数据库。

·许多城市或州的卫生部门发布了关于餐饮饭馆检查的数据，它是关于城市中包含哪些餐饮饭馆以及它们是否干净的一个很好的免费数据源。

·很多机构比如联邦医疗保险(htp：//medicare.gov)和美国食品和药物管理局(htp：//www.fda.gov)提供了大量的可下载的关于药品的用途、成本和用法的数据。

·在线留言板通常提及公司、产品和场所以及可以从中挖掘情绪和人际关系的文本。你会发现，虽然这些资源很多来自完全不同的地方，但是它们谈论的却是非常相似的东西。这是我所探索的问题的本质——当两个数据库存储相同的东西时，如何识别？正如您将在本章的剩余部分所看到的，这是一个很难的问题，但是解决这个问题可以挖掘带来很多令人兴奋的各种可能性。

连接数据的可能性

早在20世纪80年代，我看了一部名叫《Wall Street》的电影，有一个场面始终给我非常深刻的印象：由Charlie Sheen扮演的一个年轻的股票经纪人给他后来的导师（由Michael Douglas扮演）提供了一份非常有先见之明的股票情报。在该情报被证实非常精确之后，Douglas告诉Sheen说他知道该公司工会的主席是Sheen的父亲。这意味着Douglas手下的研究人员可以发现人们和公司的关系，而在那时，它促使我思考数据连接在一起可以做什么事情。

当然，在以上场景下，这听起来让人有点毛骨悚然，但这正是像证券和交易委员会(Scurities and Exchange Commission,SEC)这样的机构为了发现欺诈和内幕交易，通过手工所做的研究。暂时抛开个人隐私问题和如家庭关系这样的个人数据，而是考虑如果几百个公共数据源能够结合在一起，我们可以发现不同事物之间的关系，那将会发生什么事情呢？我们会发现什么？

以下是一些即兴的想法来启发你。你很可能对实施这些想法没有任何兴趣，但是希望这些想法可以引导你产生自己的关于数据连接的想法。

使用商标数据，我们可以确定哪些公司为不同的品牌服务，我们可能会结合美国农业部(Uited States Department of Agriculture,USDA)的营养学数据，以便确定哪个公司生产最多的含糖饮料。我们还可以对商标

数据进行商标标识分类，查看卡通是否更经常被用于销售高热量的产品。我们还引入更多的数据，使用美国环境保护局(EA)的数据来查看不同地方企业的污染量，以及这些数据和公司产品的销售情况的相互关联关系。

把地理数据库如Geonames和社交网络结合起来，我们可以判断人们的地理位置和距离对他们成为朋友的可能性所产生的影响。把这些影响关系和人口普查数据结合起来可以告诉我们人们成为朋友是否受到地理位置或者人口统计的影响（小城镇的人们是否关系更密切？未婚率高的人口是否与更多的社交网络使用相关？）。在政治方面，我们可以把证券交易委员会的哪个公司属于哪个行业的数据和责任政治中心(Cnter for Responsible Politics,CRP)提供的政治贡献相关的数据结合起来。这可以帮助我们确定哪个行业给哪个政治党派捐赠最多。图20-1显示了一些饼图，用于证明这种特定方式的数据混搭。



图　20-1：证券交易委员会的数据和责任政治中心提供的政治贡献相关的数据的饼图（见彩图70）

我甚至还没有谈及股票价格和留言板上的情绪分析之间的关系，试着把遗传和药物数据结合起来，或者判断在收入低的小区的餐馆是否卫生更差（根据健康检查员的报告），但是这只会使你对不同数据源连接

起来后带来的可能性有略微的了解。遗憾的是，数据集之间的自动连接很复杂，甚至接近不可能。在上个例子中，我们获取证券交易委员会的数据，它根据名字列出各个公司，但是为了在证券交易委员会找到公司，我们需要中心索引关键字(Cntral Index Key,CIK)。更进一步，像Exxon Mobile这样的公司通常不是分类为“能源”公司，而是更具体的“石油勘探”公司，因此，为了发现所有的能源公司，我们需要建立哪个公司属于哪个行业子集的层次分类。

企业内部

我们认为对Web上的数据源进行整合是一个很大的挑战，但是这种挑战的缩影却是无处不在。通常，大公司内部有一些数据库存储了相同的数据项，但是这些数据库之间却无法进行查询或者员工都无法知道他们感兴趣的数据就存储在与他们一起工作的某个同事所维护的数据库里，这些问题让人震惊（至少当我注意到这个问题时）。这个问题通常被称为“信息仓库问题”，指出信息被清晰地分隔开来，而且绝大多数情况下是不可访问的——像仓库里的一粒稻谷（我一直认为隐喻可以达到涵义延伸的效果）。

当我在生物技术行业工作，与很多制药公司探讨它们的数据集成方案时，这个问题显得非常突出。在很多情况下，公司的管理结构被分为医疗领域（重点研究疾病家族）。这些领域的人们可能采用特定的目标蛋白质集来查找药品或者寻找遗传标记来预测某种药物是否有效，他们所有的时间都在执行成本昂贵的实验，在这些基因、蛋白质及其化合物上构建大的知识集。

同时，公司其他部门的同事，或者可能提早完成项目的研究人员，通常研究或者已经研究了相同的或者相似的基因、蛋白质及其化合物，因此可能错过了一些重要的深刻的见解和重复的实验。

想想为什么会有这样的问题：即使假定所有人都同意了某种模式和查询机制，也不能保证人们会使用相同的术语来描述他们的实验。应该

使用什么字段以及如何搜索？比如“肺癌”实验而另一个人却把它描述为“腺癌”。很多工作组都已经尝试创建受控的词汇和固定的模式使得更容易发现实验，但是目前为止没有一个人完全真正破解这个问题。

生物技术实际上走在了前列，因为它至少已经明确了该问题，而且做出了严谨的工业范畴内的努力来解决这个问题。而另一方面，最近美国的银行投资的失败问题则很突出，没有人知道他们的股票经纪人的立场是什么，而股票经纪人自己也无法知道他们是否对房间里的某个人持反对立场。如果数据连接太少是一个问题，那么连接不应该连接的事物则会成为更大的问题。举个例子，政府已经做出了一些万众瞩目的案例，比如错误地确定某些人是恐怖分子或者拒绝某人登机仅仅因为他们和已知的被怀疑的人有相同的名字。

这些当然不是局限于大公司或者特定行业的信息。即使是小公司，也存在使客户和员工资料结合起来的问题。

连接数据的障碍

希望你已经相信把不同数据源的数据整合起来有很大优势。但是存在一些原因，使得人们并没有做这件事……

展现问题

试图连接数据集所面临的最基本的问题可能是很多数据存储在非常不灵活的结构中。首先，在科学和商业中，有惊人数据量的数据就存储在Excel表单，保存在人们本地的计算机中，其他人无法访问，而且不是为整合而做出的设计。

即使在数据库可以访问的公司，数据通常是存储在关系数据库中，很多都是预先定义适合最初设计时认为重要的模式。图20-2显示了餐馆数据的关系模式的一个简单例子。这对于大的、可预测的数据集很好，因为当配置调整很少时，关系数据库的性能非常卓越，但是当应用需要新的数据类型、新的字段或者需要经常添加新的关系时，这些都对关系数据库提出了难题。

我见过人们通过很多种方式来解决这个问题，但是有两种方式很突出，主要是因为这两种解决方法刚好相反。传统的解决方式是不断地重构数据库，创建新表，给已有表增加新的字段，这个过程可能代价很高，很容易犯错误且很慢。（由于篇幅问题，我在这里就不再赘述其他例子，但是每当我谈到这个问题时，总有很多人点头。）这种解决方式使得模式变得越来越复杂，最终使得整个模式变得过于复杂，难以可视

化和理解。

我见过的另一种解决方式是简单地构建一个非常基础的数据库模式，它可以支持任何类型的数据。通常的实现方式是有一个实体表、一个关系表以及如图20-3所示的UML图。

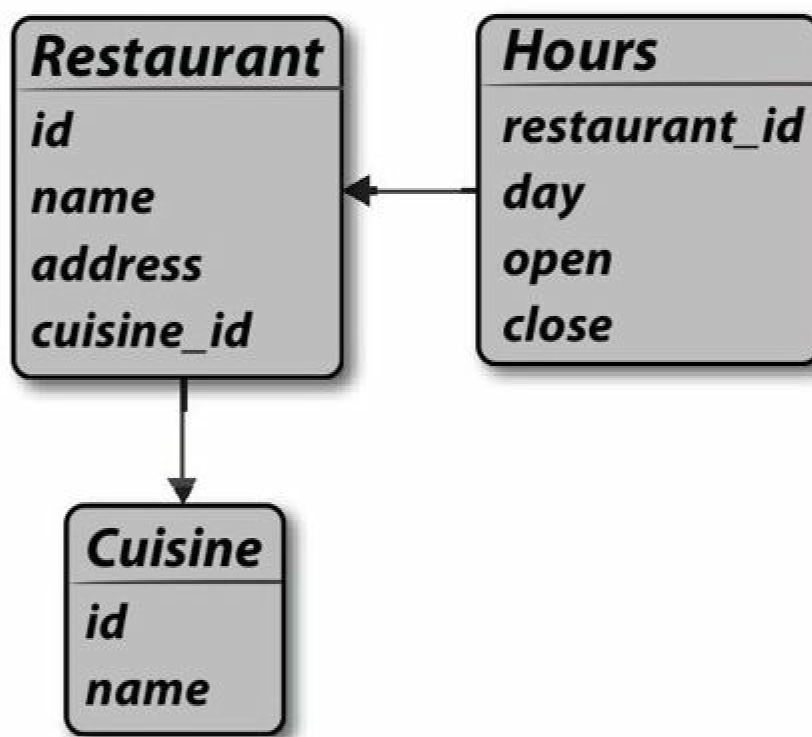


图　20-2：餐馆数据的关系数据库模式

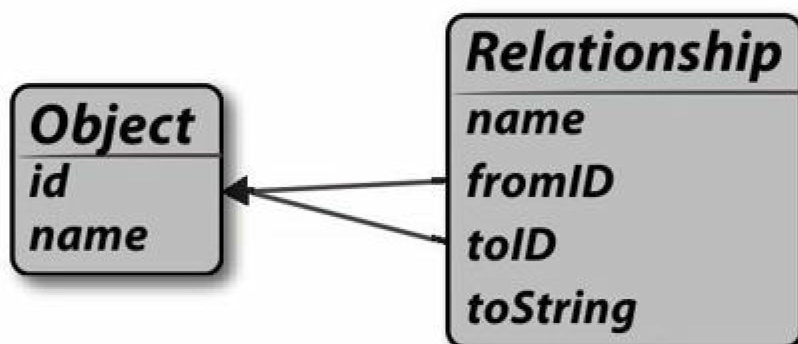


图　20-3：基础的数据库模式

这种做法有个优点，开发人员和数据加载人员可以随时给数据增加新的关系。一般来说，它把数据表示成图，而不是一组表。图20-4显示了以图形表示的餐馆，以及人们日后可以很容易增加的额外数据。

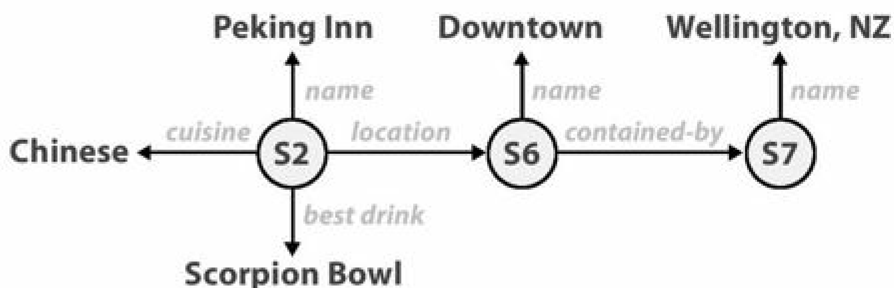


图　20-4：餐馆数据的“图形数据库”，可扩展

遗憾的是，关系数据库不是为了以这种方式来存储数据而设计的，而在该模式上做任何有用的查询将需要很多自连接操作，因此会变得非常慢。

由于这是一个非常通用的模式，因此最近几年出现了很多商业的和开源的“图形数据库”，专门为了存储这种数据而设计。以下是一些例子：

Sesame(http：//openrdf. org)

一个开源的图形数据库，由荷兰的一家软件公司Aduna维护。

Jena(http：//jena. sourceforge.net)

另一个开源的图形数据库，由Hewlett Packard开发。

AllegroGraph(http：//agraph. franz.com)

它是一个特征更丰富的商业图形数据库，由Franz开发。

Neo4J(http：//neo4j. org)

它是一个有商业授权的开源图形数据库。

虽然它们代表了在数据建模和开发实践上的转型，当从多个数据源中连接数据时，图形数据库变得更加灵活。由于这个原因，这两种数据库最近获得了很多关注，因为人们开始考虑“复杂度扩展”以及规模扩展。遗憾的是，这个属于我们连接数据时面临的最简单的问题——已经变得越来越糟糕。

共享名词和共享动词

假定你有一个非常优雅的图形或者一个完全合适的关系模式来合并两个数据库，你如何知道哪些数据项真的相互匹配？比如可口可乐公司，一个数据库可能是“Coca-Cola”，而另一个是“The Coca-Cola Company”，或者更简单的“Coke”（但是这些都和“The Coca-Cola Bottling Company”没有关系，它是独立区分开的）。

更糟糕的是，人们在命名各个字段或者对象属性时，几乎没有任何一致性可言。在一个数据库中，一个餐馆地址可能通过address字段表示，而另一个数据库可能通过location字段表示。我们如何识别一个数据库的location字段在作为餐馆的一个属性时，表示的是地址，但是当location用于描述基因序列时，表示的是在染色体中的位置。在实践中，这通常是一个人工识别过程，但是如果我们期望构建一个系统，可以很容易整合成千上万的数据，我们需要找到一些方式来消除整合过程中需要涉及的人工操作。

人们已经尝试了很多种办法来解决这些命名问题。在语义Web社区产生了“链接开源数据”，人们鼓励其他人通过标准的通用资源标识符(UI)来表示特定的对象（如一部电影、一个人或者一个餐馆），因此当两个人探讨相同的事物时，所有人都可以理解。人们已经为一组本体(otologies)进行标准化，这些本体描述了应该使用哪些字段来描述在所有情况下的餐馆或电影这类事物。

然而，到目前为止，同意使用相同的URI来表示事物以及采用相同的本体论来描述事物的组织在所有的免费数据库中只占了非常小的比例，而且几乎没有覆盖任何公司的私有数据库。在很多情况下，即使那些试着参与链接的开源数据，当前也并没有使用相同的URI来表示显然是相同的事物。这意味着，对于像我们这样试着连接数据集，我们将需要设计自动确定两种事物相同的方式。

同一事物的不同名字

和很多人一样，当我第一次试着连接数据集时，我想到了一个不错的一级假设，确定两个事物相同的最佳方式是它们有相同的名字。我甚至认为通过使用字符串距离和子串匹配就能够很有技巧地解决像“Coca-cola”和“The coca-cola company”这样的问题。这在很多情况下适用，但是在最有意思的情况下通常失败了——而且是你最感兴趣的那些情况。

我在试着结合维基百科的电影数据和Netflix网站的电影数据时遇上的一个简单的例子是《Prêt-à-Porter》。维基百科用该名字来描述这部电影（虽然实际上它是一部美国电影），但是在Netflix上，则是用英文名

字《Ready to Wear》来描述。在你认为我们应该先把名字翻译成英语，再进行字符串比较时，首先应该注意的一点是存在很多电影，同一个语言有多个名称，如《B.U.S.T.E.D.》和《Everybody Loves Sunshine》，或者《Point of No Return》和《The Assassin》——这些电影或者是在不同国家有不同的名字或者包含和发布标题不同的工作标题。

因此，如果我们不依赖于搜索相似的字符串，我们如何匹配电影？原则很简单，而细节却非常困难（是很多个人和学术研究的主题）。举个例子，我们有两部电影，都是在1994年发布的，都是由Robert Altman导演，Julia Roberts和Sophia Lauren主演：它们有可能会是不同的电影吗？正如实际情况所示，只有一部电影满足这些特征，因此包含这些属性的任何电影——不考虑它的名字——一定是相同的电影。在本章后面我将探讨在实践中是如何发现这一点。

顺便说一下，字符串距离匹配很不适合电影匹配。《Ghostbusters》和《Ghostbusters 2》是非常相似的字符串，但是它们是两部不同的电影，你可以很容易发现一部在1984年上映而另一部在1989年上映。甚至无法假定电影标题后面的数字表示的是一系列电影——《The Madness of George III》和《The Madness of King George》实际上指的是同一部电影。

同一名字的不同事物

没有意识到两种事物实际上相同通常会造成很多麻烦，生成一些副本，必要的话可以在事后解决。更严重的后果是由于两个不同事物有相

同的名字或者一些其他不足以确定“身份”的属性而导致不正确地确定它们是相同的事物。这种方式更危险的原因是一旦数据库中的事物不正确地合并在一起，在不需要太多人工介入的情况下，我们就没有很简单的方式来解决掉这些问题。

在威斯康星州（美国）有7个城镇名字叫“富兰克林”，其中只有一个城镇有沃尔玛。至少有四本书的名字为《City of God》。至少有50个名叫“John Smith”的人有足够的知名度出现在维基百科上。抛开考虑错误的飞行黑名单，即使两个事物具有相同的名字，很显然也不足以证明它们是相同的，尤其对于人们的名字而言。虽然对于人来说，存在唯一关键字，如在美国是社会保险号码，但是这些信息几乎从来不会出现在公众可访问的数据库上；对人的唯一识别符通常需要某种程度的保护。

除了在非常封闭的集合（比如国家名字）或者是非常罕见的名字（我想到了“Toby Segaran”），强烈建议不要仅仅基于名字来合并事物，应该使用额外的信息来设计算法，以便确定两条不同的记录确实是同一个事物。

我觉得必须指出已经付出很多努力来为某些特定事物创建经典的标识符，但是这些标识符由于隐私问题从未被成功地应用于人。在美国，我们有社会保险号码，很多政府部门和信誉机构用它来追踪我们，但是人们告诉我们不应该把这些信息和别人分享，因此我们当然不会把它放到公共数据库中以防别人可以更容易地把他们的数据和我们的数据链接起来。因此，我们永远都处于几十亿的人们有相似的名字而没有办法识

别出他们。

可能的解决方案

在通常情况下，能够意识到这是一个悬而未解的问题很重要，人们尝试了一些在某些特定情况下可以工作的想法。这些方法，有的是个“死胡同”，但是有的当继续发展时，看起来像是可以在广泛的数据集上工作。

匹配多个字段

在第7章中，Jeff Jonas描述了一个假设的场景，可以通过结合名字和地址，发现某个员工是个小偷。在那种情况下，名字和地址的组合足以表明两种不同的记录实际上表示同一个人。Jeff可以很快地指出他遇到这样的情况，一个“Patrick Smith”和另一个“Patricia Smith”有相同的地址，而且都包含“Pat Smith”，因此如果你不注意，很容易陷入其他很明显的规则的迷宫中。

这说明了在数据集中匹配数据项时可用的最基本常见的方式：选择一组参数，创建一组固定的规则来告诉你事物间是否匹配。举个例子，“两个人是否有相同的名字和地址？”或者“两部电影是否有相同的名字并且是在同一年发布的？”

这种方法在很多情况下都适用，但是它存在一些缺点。首先，它需要开发人员识别字段和匹配规则。这种方式可能会非常乏味，因为当他们发现根据基本的名字/发布年份规则，《Prêt-à-Porter》和《Ready to Wear》不匹配时，他们需要发明另一种规则，如“两部电影在同一年发

布，导演相同，而且至少有一个演员相同。”

另一个问题是它需要这些字段本身具有很高的一致性。如果我们没有演员的全名怎么办呢？如果在其中一个数据库丢失了某些电影的年份信息，那又该如何处理？最后，因为我们是选择特定的字段，然后生成平面记录，这种方式并没有充分利用全部网络的数据——全网数据潜在地可能为我们提供更多关于身份的信息。

集体调解

我相信充分利用全网数据是解决匹配问题的关键。这种思想体现在称为“集体调解”(cllective reconciliation)或“集体实体决议”(cllective entity resolution)。有关这方面的详细讨论，我建议阅读Indrajit Bhattacharya的博士论文，你可以在http：//www.lib.umd.edu/drum/handle/1903/4241获取该论文。

在本节中，我将从高层次介绍集体调解的涵义。这些算法的实现细节根据你所处理的特定类型的数据会有很大区别，而且超出了本章的讨论范围，但是我希望一个高层次的概要会帮助你去试验以及使你阅读别人在该主题上所做的工作会更容易。

首先，考虑如下情况，我们有两个电影数据集，两个数据集包含的信息略有不同。从这两个数据集抽取两份数据片段并进行图形化表示，如图20-5所示。我们已经确定仅仅基于名字进行匹配是不明智的，因此我们无法仅仅通过名字来确定两个对象是一样的。但是，我们相信图A中的节点node10和图B中的节点node22可能是一样的，因为它们的名字都是“Julia Roberts”，图A中的节点node12和图B中的节点node27可能是一致的，因为它们都是命名为“Ready to Wear”。

技巧在于我们在两个图中的数据项有潜在的匹配，而且这些数据项相互间有一定的连接——潜在匹配的“Julia Roberts”节点和潜在匹配的“Ready to Wear”之间有连接。这条连接为这二者的匹配提供了更多的

证据。这种匹配关系是否是最终结论取决于很多可能性假设，如有多个名叫Julia Roberts的女主角在名为《Ready to Wear》的电影中，但是在该案例情况下，我们可以认为这些关联匹配是正确的。

这些关联的实现方式有很多区别，但是主流技术称为“消息传递”(mssage-passing)。通常，图A中的节点node12知道它可能和图B中的节点node27一样，查看图B中的所有连接，它给图A中的邻居节点发送了消息。连接所有演员的消息可能是“你可能和节点node22或节点node25一样”。而节点node10接收到这条消息后，意识到“实际上，我已经认为我可能和节点node22一样”。同理，节点node10告诉节点node12它可能的所有电影。图20-6显示了可能的过程。

你很可能明白了为什么这被称为“集体调解”，而不是拼合记录，我们实际上是想要一次性把所有东西都归并起来，而且节点之间可以互相提供信息是否需要归并。当然，这只是一个很小的例子，但是考虑一下图20-7所示的更复杂的一项。

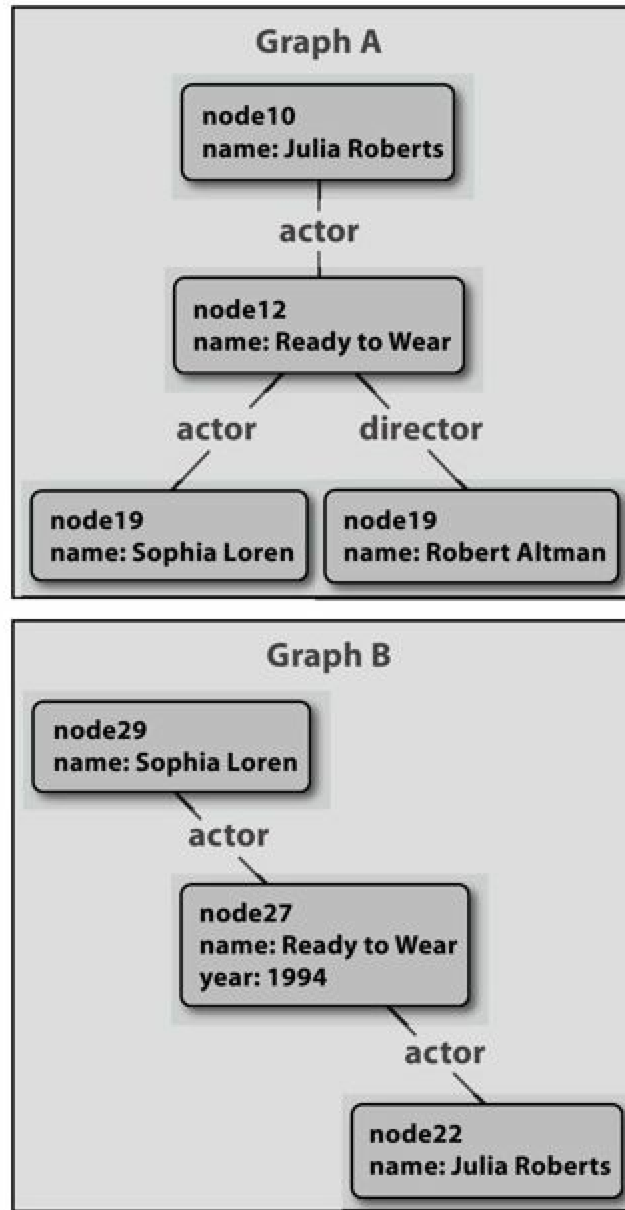


图　20-5：两个不同的电影数据集的数据片段

在该图中，名字为“Ready to Wear”的节点并不匹配，而且我们甚至没有一个节点命名为node10！我们如何知道它是Julia Roberts？但是，我们已经稍微扩展了网络范围，包含一些其他Julia Roberts为明星的电影，幸运的是，“消息传递”机制可以通过很多迭代来完成。可能你明白

了“消息传递”算法的作用？以下是基本的思想：

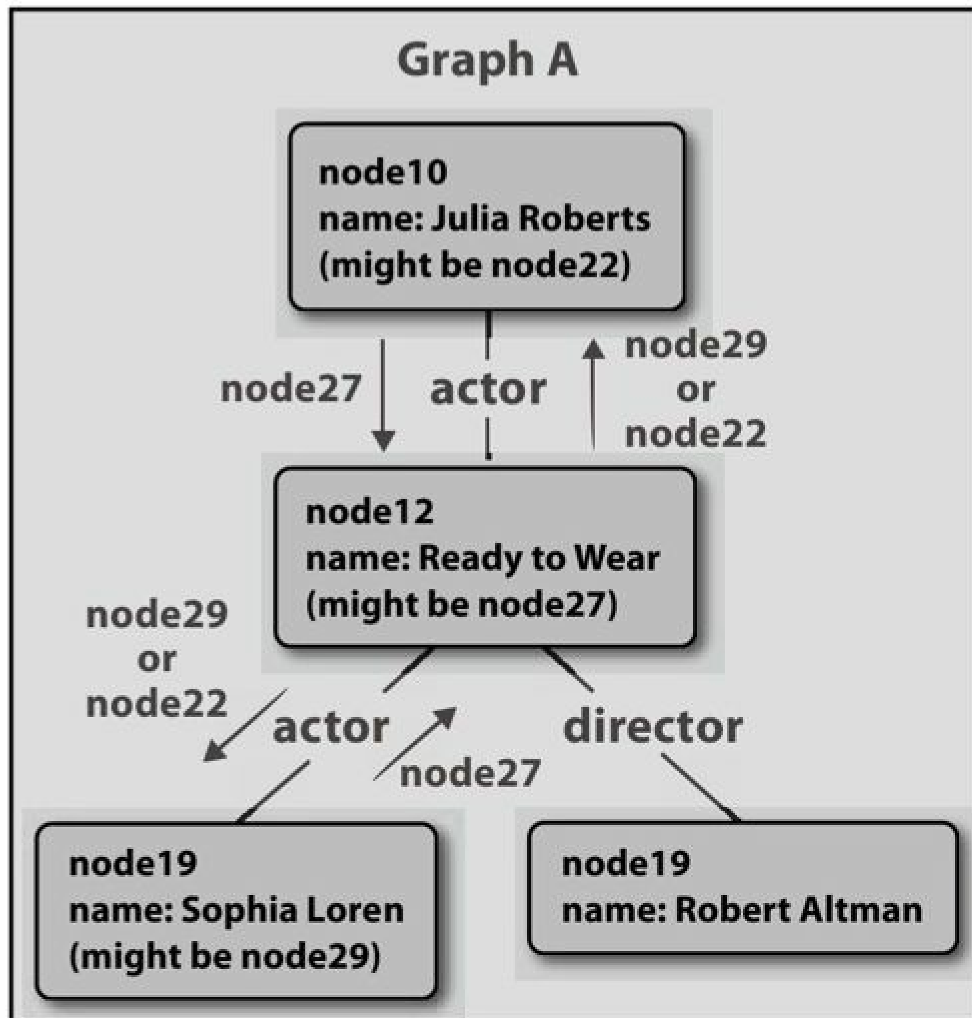


图　20-6：节点间的消息传递

1.节点node11决定它和节点node23有可能匹配，因为它们的名字相同；

2.相似地，节点node15认为它和节点node9有可能匹配（如果我们有更多关于Julia Roberts的电影数据，就可以继续更多这样的推测了）；

3.节点node11发送了一条消息给节点node10：“你可能和节点node22或者节点node24匹配”；

4.类似，节点node15发送了一条消息给节点node10：“你可能和节点node22或节点node25匹配”；

5.节点node10在收到了所有这些消息后，下定结论认为很可能是节点node22，因为那是它所共有的消息；

6.注意，节点node10已经确定了自己所匹配的节点，因而可以发送一条消息到所有它所连接的节点（包括它收到消息的节点），给出消息它所认为的它们可能和哪些节点相匹配；

7.节点node12收到了节点node10和节点node19的消息：“你可能和节点node23匹配”，因此它最后选择和节点node23匹配。

虽然只有一个演员名字相同，但是我们已经确定《Prêt-à-Porter》和《Ready to Wear》是同一部电影。

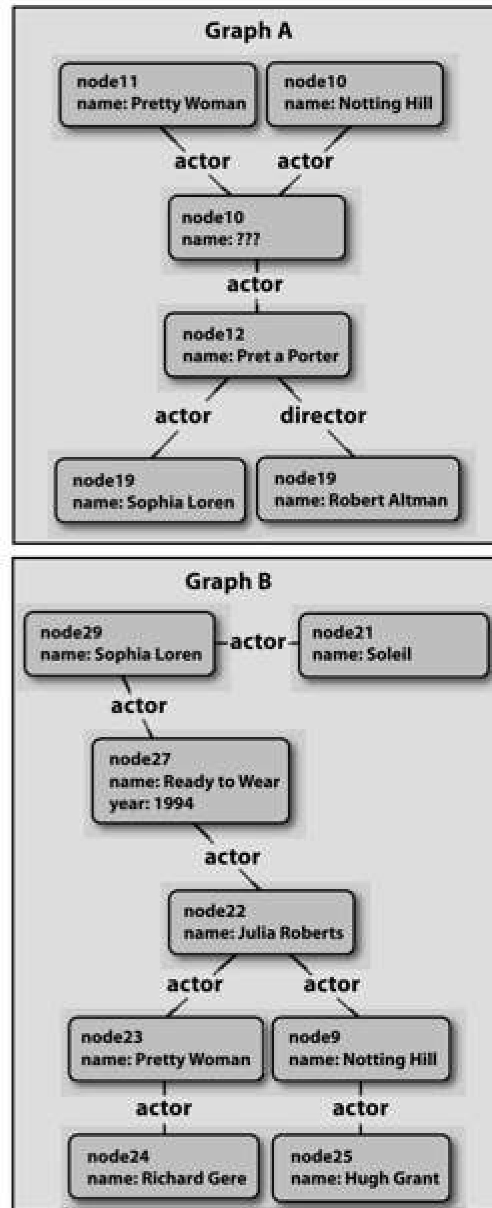


图　20-7：一个更复杂的归并问题

注意我们是如何利用全网的事实来确定最终的匹配？这就是集体调解的精髓，以及它如此强大的原因。这一观点可以继续扩展：在某些实验中，我发现你可以仅仅使用电影的发布年份来连接两个电影和演员的数据集。某个明星在特定12个年份出演的电影，以及和在8个特定年份

另一个不同的明星共同出演的事实就足以唯一性地确定这两个人的身份。

当然，其中的实现和数学细节可能非常复杂，但那些不在本章的讨论范围之内。实现这种技术的可定制性版本就留个读者作为非常有益的练习了。

结论

到目前为止，多数人都意识到几乎每个领域都越来越依赖于数据分析的进步。虽然科学主要是依赖于从很少的观察中构建的理论，未来看起来似乎需要收集和挖掘好几百万个衡量尺度；虽然零售业公司主要依赖于“趋势观察者”提供的洞察力，但是现在很多人相信他们应该销售的东西已经淹没在大量收集的数据中了。

不是花费更大代价独立构建更大的数据集，我相信未来在于利用别人生成的大量数据，把它和我们自己生成的数据进行结合和混合。不论这些数据是否来自我们自己的企业内部、非营利结构或者公共领域，都可以通过重用和连接数据来节省很多开支。希望本章能够启发你找到更好的方法来实现这些。

附录　作者简介

Ben Blackburne是Wellcome Trust Sanger Institute的序列分析和汇编研究组的博士后。

Jean-Claude Bradley是Drexel大学化学专业的副教授以及艺术和科学学院的网络学习协调员。他带领UsefulChem项目，该项目在2005年夏天启动，目标是通过实时地把所有的研究工作发表到公开博客、wiki和其他Web页面，使得科学过程尽可能透明化。Jean-Claude发明术语“Open Notebook Science”以便和更严格的“Open Science”等其他方式区别开。他教大学生组织化学课程，其绝大部分课程内容都能够通过公开博客、wiki、游戏和音频视频来自由获取。他在组织化学专业获得博士学位，在合成和机械化学、基因治疗、纳米技术和科学知识管理领域发表过论文和获得专利。

Lukas Biewald是Dolores实验室的创始人和CEO，该公司致力于使“开放来源”(cowdsourcing)变得容易简单。Dolores实验室的博客(htp：//blog.doloreslabs.com)充满了有趣的“开放来源”技术和数据可视化实验。在创办Dolores实验室之前，他是Powerset公司的资深科学家；在加入Powerset之前，他构建了Yahoo!Japan的搜索引擎排序算法。他在斯坦福大学获得了数学学士和计算机硕士学位，他在人工智能实验室工作，并发表了两篇有关机器学习应用的论文。他的个人网站是：http：//lukasbiewald.com/。此外，Lukas还是个专家级别的围棋手。

Brian Cooper是雅虎研究院的首席科学家。在加入雅虎之前，他是Georgia Tech的助理教授。在那之前，他在斯坦福大学获得了博士学位。他的研究兴趣是构建分布式系统，尤其是通过数据库方式管理和处理数据的分布式系统。在雅虎，他的工作是构建庞大的分布式数据存储和处理系统。早期，他的工作方向是自适应的P2P系统、分布式流事件处理、可靠的分布式存档数据存储和XML索引。

Jason Dykes从20世纪90年代早期就一直设计和开发用于探索的交互式空间接口。他使用了一系列灵活的技术进行快速开发，包括使用Tcl/Tk、SVG/JavaScript和Processing来开发创新的软件应用和显示地理结构的创新视图。作为伦敦城市大学(htp：//gicentre.org)的资深讲师，1990年他在牛津大学获得地理专业学士学位，2000年在Leicester大学获得博士学位。Jason是在地理可视化方面的国际制图联委会(Iternational Cartographic Association Commission)的联合主席。

Jonathan Follett是Hot Knife Design公司的总裁和首席创意官。他在国际上发表了关于用户体验、信息设计和虚拟团队方面的作品。他在《A List Apart》、《Digital Web》和《UXmatters》上都发表过作品，给Boston区的技术组织做Web相关的主题讲座。他的文章被翻译为中文、印度语、葡萄牙语、俄语和西班牙语。Jon的虚拟设计作品获得多个美国图形设计奖、一个Horizon交互奖和其他业界的认可。

Andrew Gelman是哥伦比亚大学的统计学和政治科学教授。他最近出版的书有《Data Analysis Using Regression and Multilevel/Hierarchical

Models》（剑桥大学出版社）、《Red State,Blue State,Rich State,Poor State:Why Americans Vote the Way They Do》（普林斯顿大学出版社）和《A Quantitative Tour of the Social Sciences》（剑桥大学出版社）。

Yair Ghitza是哥伦比亚大学的政治学专业博士生，致力于研究美国政治学和量化方法。他之前在政治分析公司工作，包括Catalist公司和Copernicus Analytics公司。

Rajarshi Guha是NIH化学基因组中心的科学家，研究高吞吐量屏蔽问题的各个方面。在这之前，他是Indiana University信息学院的访问学者。在过去几年，他研究了化学信息学和计算药物发现，范围从QSAR建模和算法开发到工具箱的软件工程和部署化学信息方法和模型的Web服务基础设施。

Alon Halevy他领导一个结构化数据管理研究组。在这之前，他是西雅图华盛顿大学计算机专业的教授。在1999年，Halevy博士是Nimble Technology公司的创始人之一，该公司是第一个在企业信息集成空间领域的公司；在2004年，他成立了Transformic公司，为Deep Web发明搜索引擎，该公司后被G公司收购。Halevy博士是Computing Machinery组织的会员，在2000年获得为科学家和工程师颁发的Presidential Early Career Award(PECASE)；他是Sloan会员（1999～2000）。他发表了150多篇技术论文。他于1993年在斯坦福大学获得了计算机专业的博士学位。

Jeff Hammerbacher是Cloudera公司的产品副总裁和首席科学家。在加入Cloudera之前，Jeff是Accel Partners公司的一个企业家。在加入Accel

之前，他在Facebook构思、创建和领导了数据组。在Facebook，数据组是负责驱动很多统计和机器学习应用，以及构建为大数据集支持这些任务的基础平台。该数据组发表了一些学术论文和两个开源项目：Hive是在Hadoop上构建的离线分析系统；Cassandra是在P2P网络上的结构化存储系统。在加入Facebook前，Jeff是华尔街的定量分析师。Jeff在哈佛大学获得数学专业的学士学位。

Jeffrey Heer是斯坦福大学计算机专业的助理教授，他主要研究人机交互、交互可视化和社会计算。他的工作成果包含探索数据的创新性可视化技术，简化可视化创建和定制软件工具以及为最大化利用多个分析员的洞察力的协作分析系统。他是开源可视化工具箱prefuse和flare的作者，这两个工具当前被可视化研究组织和很多公司采用。在过去几年里，他也曾就职于Xerox PARC、IBM Research、Microsoft Research和Tableau Software。他在加州大学伯克利分校获得计算机专业的学士、硕士和博士学位。

Matthew Holm是波斯顿的Hot Knife Design公司的咨询创造主管，工作主要涉及公司的战略、HTML/CSS的开发以及CMS驱动的Web站点。Matt当前是Oregon的人机交互论坛的副总裁(te Oregon chapter of the Association of Computing Machinery's Special Interest Group on Computer-Human Interaction,CHIFOO)。除了在线网络领域的工作，Matt还是专业的儿童书籍作者和解说员。他的作品图片小说《Babymouse》(Rndom House出版社）备受赞誉且获得奖项，目前已经印刷超过100万册。

J. M.Hughes是一位嵌入式系统和软件工程师，对实时控制、数据采集和图像处理特别感兴趣。从2003～2007年，他负责设计、实现和测试火星着陆器凤凰号的表面成像软件。他当前致力于多波长的激光干扰仪系统的电子和控制软件，该系统可以为NASA项目验证望远镜镜片分割。他和妻子、女儿一起住在亚利桑那州的Tucson。

Jeff Jonas是IBM首席科学家、IBM实体分析组(IM Entity Analytics Group)和杰出工程师。IBM实体分析组是基于系统研究和开发的技术成立的，由Jonas在1984年创办，2005年被IBM收购。Jonas的博客是http：//jeffjonas.typepad.com。

Jonathan P. Kastellec是普林斯顿大学的政治学教授。他在《Journal of Law》、《Economics＆Organization》、《Journal of Empirical Legal Studies》和《Perspective on Politics》发表过论文。

Valdean Klump目前居住在旧金山，是一名作家，在一家创意实验室工作。

Aaron Koblin是一名来自旧金山的艺术家，他的可视化数据项目Flight Patterns、The Sheep Market和Ten Thousand Cents都广为人知。他是视频“House of Cards”的技术主任，当前是一家创意实验室的设计技术带领人。

Coco Krumme是MIT Media Lab的研究生。她同时也为在旧金山的Metaweb Technologies公司工作。

Andrew Lang是Oral Roberts大学的数学教授。他的博士研究领域是弯

曲时空的量子理论。在该领域他还很活跃，同时他也喜欢参与跨项目的协作工作，范围从篮球自由投篮建模到在冲击力下的旋转宇宙飞船的稳定性。他当前的研究兴趣包含多维数据可视化、科学和科幻小说的关系、技术和形而上学自然主义在认识论上的区别。

Pierre Lindenbaum 2000年获得病毒学的博士学位，当时他研究virushost交互。后来他转行研究生物信息学，在French National Center of Genotyping工作一年后，他在2001年加入了French startup Integragen。他现在作为生物信息学家，在巴黎的遗传研究中心Fondation Jean Dausset-CEPH工作。

Jayant Madhavan是一名资深软件工程师，是Deep Web爬虫项目计划的技术带领人。在这之前，他是Tranformic公司（2005年被G公司收购）的首席架构师，Tranformic公司为Deep Web发明了搜索引擎。Madhavan博士2005年在华盛顿大学获得计算机专业博士学位。

Michal Migurski是Stamen Design公司的合伙人，同时是该公司的技术和研究方面的带领人。他从1995年开始就一直构建Web，专于处理大规模的、振奋人心的数据集，以及这些数据和大量的客户端的用户进行通信和分发的方式。他公开为学术和工业领域做一些主题讲座，积极参加各种开源开发项目，积极维护Web博客：http：//mike.teczno.com。他在加州大学伯克利分校获得学位。

Cameron Neylon是在跨领域工作的生物物理学家，著名的开源研究实践和改进的数据管理的倡导者。他当前作为生物分子科学的资深科学

家，在Science and Technology Facilities Council(STFC)的ISIS Neutron Scattering facility工作。在学术研究上，他经常在科学Web技术和成功的（和不成功的）通用和设定设计工具的应用方面发表论文或进行演讲。

Peter Norvig他是AAAI和ACM的会员，合著了《Artificial Intelligence:A Modern Approach》(Pentice Hall出版社）一书，这是在人工智能领域的一流的教材。在这之前，他是NASA计算机科学的负责人、USC和Berkeley的教师。

Brendan O'Connor是机器学习和自然语言处理的研究员。他是Dolores实验室的科学顾问，先前作为相关性技术工程师在Powerset公司工作。他从斯坦福大学获得符号系统学的学士和硕士学位，目前回到学术领域，作为研究生去卡内基梅隆大学就读。他的博客名为“Artificial Intelligence and Social Science”，可以访问：http：//anyall.org/blog。

David Poole在AT＆T实验室的统计研究部门工作，目前是美国统计协会(Aerican Statistical Association)的统计计算部门秘书和财务主任。他在大规模的数据挖掘方面很有经验，如通信工程的客户呼叫数据分析和欺诈识别。

Raghu Ramakrishnan是雅虎Audience and Cloud Computing的首席科学家，是一名研究员。他在数据库系统的工作——主要研究数据挖掘、查询分析和Web规模的数据管理——影响了商业数据库系统的查询优化和SQL：1999的窗口函数的设计。他的Birch聚类算法论文，获得了SIGMOD 10-Year Test-of-Time奖，他还著有被广泛采用的教材

《Database Management Systems》（和Johannes Gehrke合著；McGraw-Hill出版社）。他是ACM SIGMOD的主席，ACM和IEEE的成员。

Toby Segaran是O'Reilly的两本很受欢迎的书：《Programming Collective Intelligence》和最近出版的《Programming the Semantic Web》的作者。他当前在Metaweb工作，在那里他开发了大规模的协调算法，目的是为所有其他公共数据库创建包含共享关键字的免费数据库。在加入Metaweb之前，他成立了生物技术软件公司，2003年被Genstruct收购，Genstruct是系统生物公司。Toby在MIT获得了计算机学士学位，现在和妻子Brooke一起居住在旧金山。在他的博客上你可以读到更多关于他的文章和数据实验：http：//blog.kiwitobes.com。

Lisa Sokol当前是IBM全球商务服务组的顾问，专门做实体分析。她的主要研究领域是帮助执法人和情报组织发现深埋在大数据集中的“可采取措施的信息”(Ationable Information)。她架构了大量的系统来检测评估相对于欺诈、恐怖主义、反间谍和犯罪活动的威胁风险。她还帮助开拓了一些技术的应用，如数据挖掘、文本挖掘和机器翻译，用来探索共享情报环境中的可访问信息。Sokol博士在这些领域发表了无数篇论文。她在Massachusetts大学获得操作研究的博士学位。

Utkarsh Srivastava是雅虎研究院的资深研究科学家。他的主要研究领域是构建系统来解决大规模数据管理系统。在开发PNUTS之前，他在开发Pig项目中发挥了重要作用，Pig是在Hadoop之上的描述查询语言。他在斯坦福大学获得了博士学位，在那里他的研究工作是流数据的查询处

理和一些查询优化问题。

Deborah Swayne在AT＆T实验室的统计研究部门工作，是美国统计协会的成员，统计图形学的ASA部门的前主席。她是已经被广泛使用的多维数据可视化软件ggobi的作者之一。

Jud Valeski是Gnip公司的CTO和创始人之一。Gnip公司致力于开发数据便携式软件。从客户端用户接触的产品到大规模后台基础设施项目，Jud工作了20多年，并乐在其中。他在IBM、Netscape、onebox.com、AOL和me.dium加入过工程师、产品和M＆A团队。在发布的由几千万的用户使用的产品中，Jud发挥了重要的作用。

Hadley Wickham是Rice大学的助理教授，研究兴趣是开发工具（包括计算和认知两个方面）使得数据准备、可视化和分析变得简单。他开发了15个统计工具R的工具包，在2006年，由于在ggplot软件上的工作以及重塑了统计工具R的工具包，Hadley获得了统计计算的John Chambers Award。

Antony Williams是ChemZoo公司的总裁，提供ChemSpider服务，为化学家提供在线免费访问服务，目的是为科学家构建以结构为中心的组织。他在商业科学软件领域的Advanced Chemistry Development(ACD/Labs)当了10多年的首席科学官，任期中监督产品开发、市场和销售团队。他是个成就很高的NMR光谱学家，发表了100多篇论文。他曾经是Eastman Kodak Company的NMR技术带头人，在学术领域和政府研究所都工作过。他最近热衷于提供ChemSpider服务，为大

众提供化学相关的信息和软件服务。

Egon Willighagen是瑞典Uppsala大学的科学家，主要研究制药生命科学领域的数据分析。他主要研究设计统计方法的开发和生物分子化学计量学和蛋白质化学计量学。他是Chemistry Development Kit和Metware的发布主管，在其他化学信息学开源项目工作了10多年，其中两个项目是Jmol和Bioclipse。

Jo Wood是伦敦城市大学giCentre(http：//gicentre.org)地理信息学的读者。他从1990年就一直致力于山水地貌的分析，在GIS上发表了《LandSerf》，用于进行表面视觉探索。作为地理学家和程序员，他在过去几乎10年的时间中一直在用Java开发软件为数据分析问题找到地理可视化的解决方案。他的著作有《Java Programming for Spatial Sciences》(CC)。当不用计算机分析地形地貌时，他经常骑车兜风。

Matt Wood是Wellcome Trust Sanger Institute的产品软件的负责人，在那里他负责驱动该研究所的世界级的序列设备的软件。

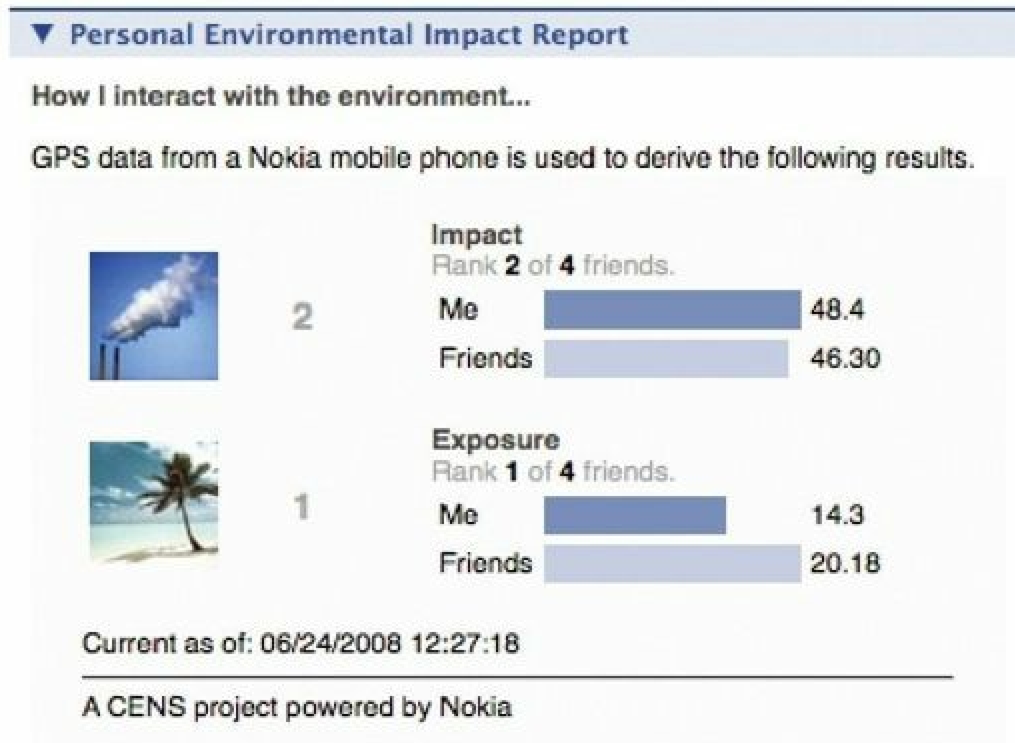
Nathan Yau是加州大学洛杉矶分校的统计学博士生，在加州大学伯克利分校获得了电子工程和计算科学的学士学位。他的主要研究兴趣包括数据可视化、自我监督以及数据本身和物理世界如何纠织在一起。深受暑期《New York Times》的图形编辑实习生工作的启发，Yau维护了一个领先的统计学和数据可视化博客：FlowingData(http：//flowingdata.com/）。该博客主要涉及设计家、统计学家和计算机科学家如何使用数据帮助我们做出更好的决策。



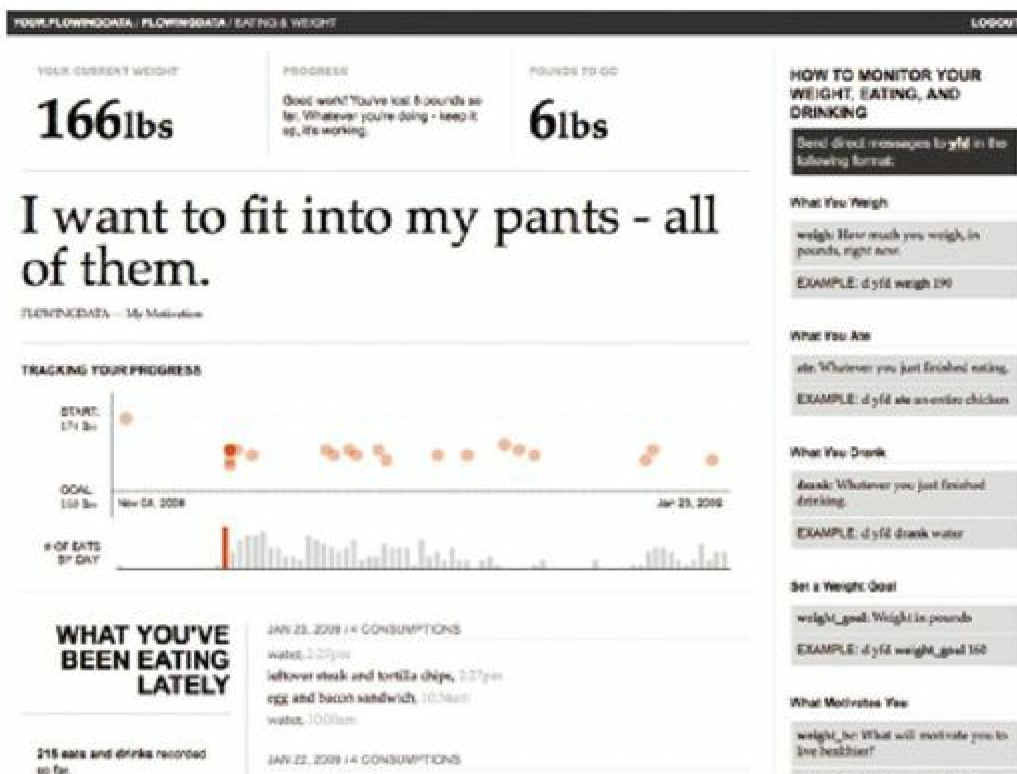
彩图　1（图1-1）　我们在地图上对不同的视觉提示进行实验，通过影响和排放值来最佳展示地理数据。以上展示了我们初始设计的三次迭代。最左边的地图显示根据碳影响所呈现的彩色GPS轨迹；对于中间的地图，我们通过单色的圆圈面(aea　circle)来表示影响；对于右边的地图，我们结合了GPS数据来呈现用户什么时候有空，然后重新采用单色的颜色编码方案



彩图　2（图1-2）　在当前的映射机制中，我们采用颜色过滤器来高亮显示数据。地图仅仅是提供上下文信息。链接的直方图显示映射后的数据的影响和排放值分布。当用户通过滚动条看直方图的某一栏，相应的GPS数据会在地图上高亮显示



彩图 3（图1-4） PEIR的Facebook应用允许用户分享他们对影响和排放值的发现，以及与朋友的进行比较



彩图　4（图1-5）　人们为不同原因跟踪他们的体重和饮食。YFD把这些动机作为用户界面的焦点



彩图　5（图2-2）　为易读性而设计

What is your opinion of this new product?

URBAN WALLACE ASSOCIATES

Take this one-page survey and let your opinion be heard.

First, please review the summary of this new product and then take our short survey. It's that simple. Rest assured that we do not collect any personally identifiable information about you.

Step 1 Please review this new product.

Luxury Product Summary

A major US manufacturer is considering offering a new luxury product that [details removed].

- The price will be 15% lower than that of similar products.
- Each one is handcrafted by world-class artisans.
- It can be customized to your exact specifications.

Guaranteed Satisfaction

- The product comes with a **lifetime satisfaction** guarantee.
- Our customer service representatives are **guaranteed** to be on-call to assist you, 24 hours a day.

Other Specific Benefits

- The product will be offered in an extremely limited edition.
- Other products from this manufacturer have appreciated more than 50% in value over their lifetime.
- The safety features of this product have been independently rated as among the highest in the world.

Step 2 Please answer a few survey questions.

1. How interested would you be to purchase this kind of product?

Would you say that you:

- Definitely would purchase
- Probably would purchase
- Might or might not purchase
- Probably would not purchase
- Definitely would not purchase

2. Do you currently own a product like this?

- Yes
- No

3. Do you currently own a product manufactured by [competitor's name]?

- Yes
- No

SUBMIT CLEAR

彩图　6（图2-3）　调查开始只有3个问题

彩图　7（图2-4）　基于用户输入，调查可能会扩展到6个问题

Step 2 Please answer a few survey questions.

1. How interested would you be to purchase this kind of product?

Would you say that you:

- Definitely would purchase
- Probably would purchase
- Might or might not purchase
- Probably would not purchase
- Definitely would not purchase

2. Which of the following are extremely important to you?

(Check up to 3 responses)

- The price of the product
- The product's lifetime guarantee
- The quality of the craftsmanship
- The fact that it can be customized to my taste
- The prestige of owning the product
- The safety features
- The on-call customer service guarantee

彩图　8（图2-5）　调查细节——当用户对问题1回答“Yes”时

Step 2 Please answer a few survey questions.

1. How interested would you be to purchase this kind of product?

Would you say that you:

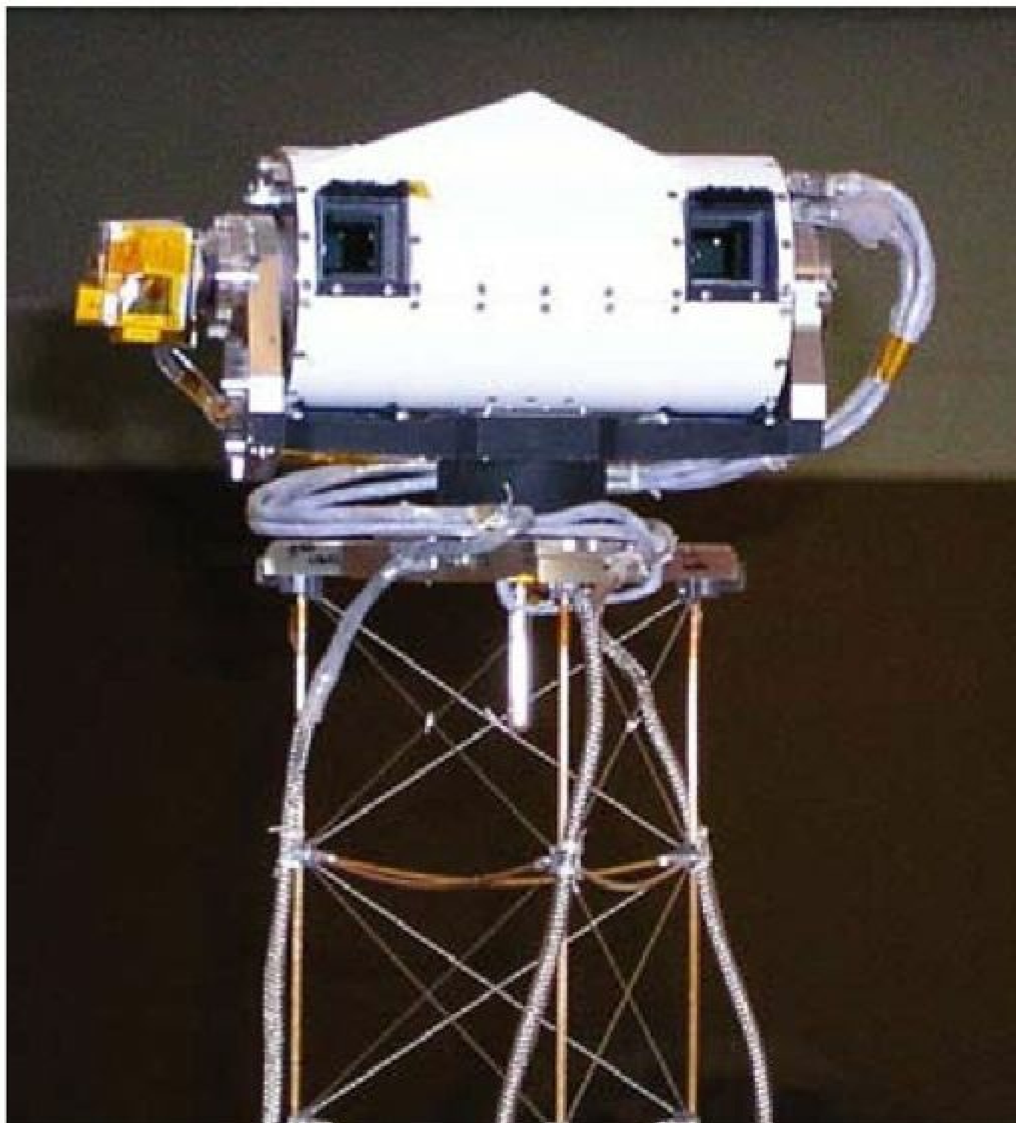
- Definitely would purchase
- Probably would purchase
- Might or might not purchase
- Probably would not purchase
- Definitely would not purchase

2. Why are you not interested in this product?

彩图　9（图2-6）　调查细节——当用户对问题1回答“No”时



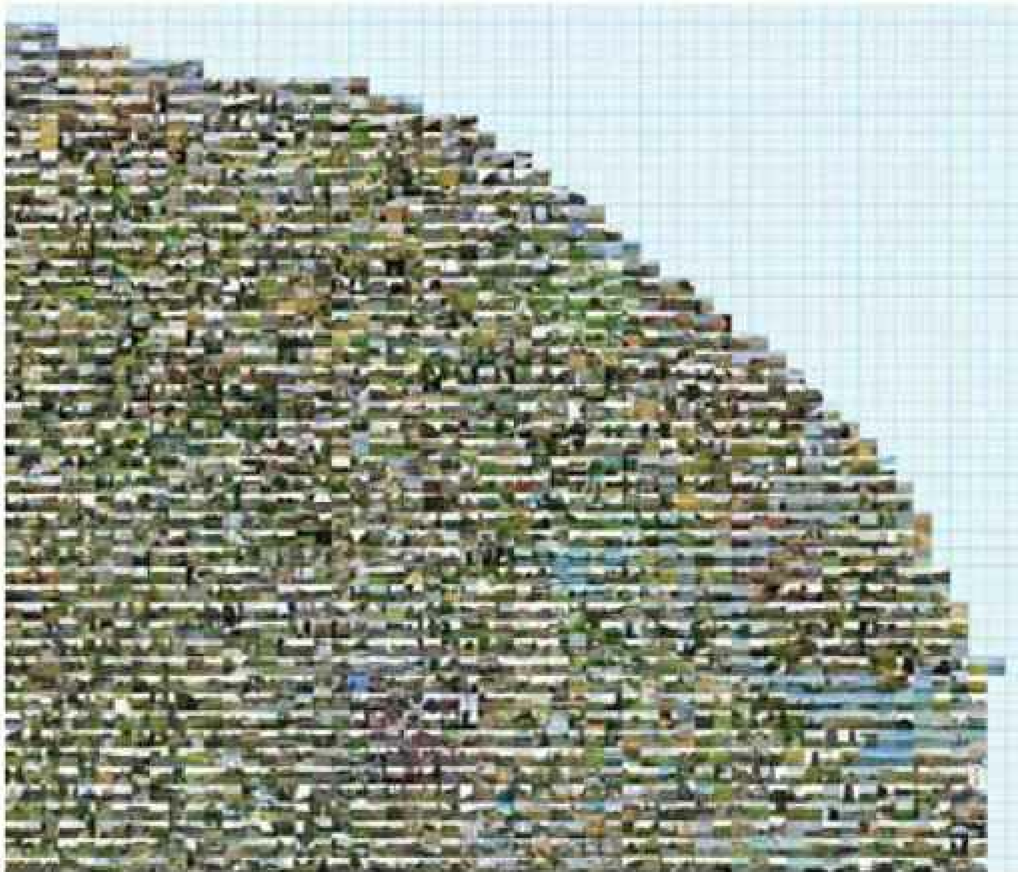
彩图　10（图3-1）　　凤凰号火星探测器在火星上的着陆效果图（图像来源：美国航空航天局和喷气推进实验室NASA/JPL)



彩图　11（图3-2）　立体表面成像器（图像来源：Arizona大学/NASA/JPL)



彩图　12（图6-1）　Aberuchill附近的小路。远处是Bioran Dalchonzie。这是Geograph项目的网站出现的第100万张图片(htp：//www.geograph.org.uk/photo/1006884）。图片版权归Dr.Richard Murray所有，由Creative Commons License(http：//creativecommons.org/licenses/by-sa/2.0/）授权使用



彩图　13（图6-2）　Norfolk海岸的地理马赛克图。每个图片都是根据其地理位置进行映射，表示了1平方公里的景观。图片由Creative Commons License(http：//creativecommons.org/licenses/by-sa/2.0）授权使用

彩图　14（图6-4）　对于六种选定的场景类型，其出现在地理标题和评论的描述的树形图。节点大小表示术语的出现频率。颜色突出了包含一种继承随机方案的场景类型/刻面(fcet)/描述符分类。布局采用“有序的方块化”方法来维持节点之间的方块形状

彩图　15（图6-5）　对于六种选定的场景类型，其出现在地理标题和评论的描述的树形图。节点大小表示术语出现频率。颜色突出了包含一种继承的随机方案的场景类型/刻面/描述符分类。布局采用“切片和切块”的方法来增强维度（上图）间的比较和“切片和切块/有序的方块化”方法来改进标注的可读性，如下部的图



彩图　16（图6-6）　“有序的方块化”树形图，其颜色是通过CIELab颜色模型的一种色彩空间来显示绝对的地理位置，在该空间中，主观感觉到的颜色区别和地理位置的差异紧密相关

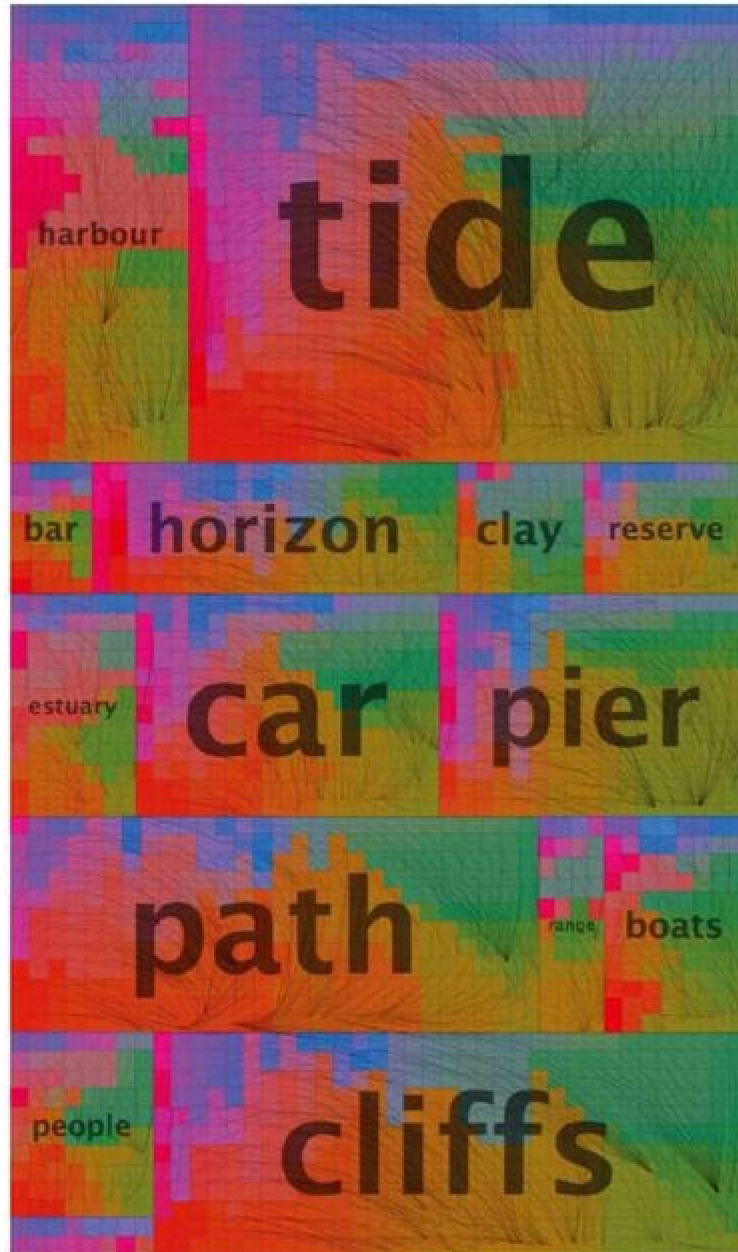


彩图　18（图6-8）　对于六种选定的场景类型，其出现在地理标题和评论的描述的空间树形图。节点大小表示词频，颜色表示使用CIELab方案的绝对空间地理位置。位移向量表示非叶子节点的绝对地理位置如场景类型、刻面和描述符

彩图　17（图6-7）　“有序的方块化”树形图，通过CIELab色彩空间的颜色显示绝对的地理位置。包含描述符节点的叶子节点，使用空间有序算法，通过地理位置做关联排序



彩图　19（图6-9）　对于六种选定的场景类型，其出现在地理标题和评论的词汇的空间树形图。位移向量表示非叶子节点的绝对地理位置（共现词），并提供了关于位移的空间聚类和空间趋势的信息来满足树形图的空间填充目标



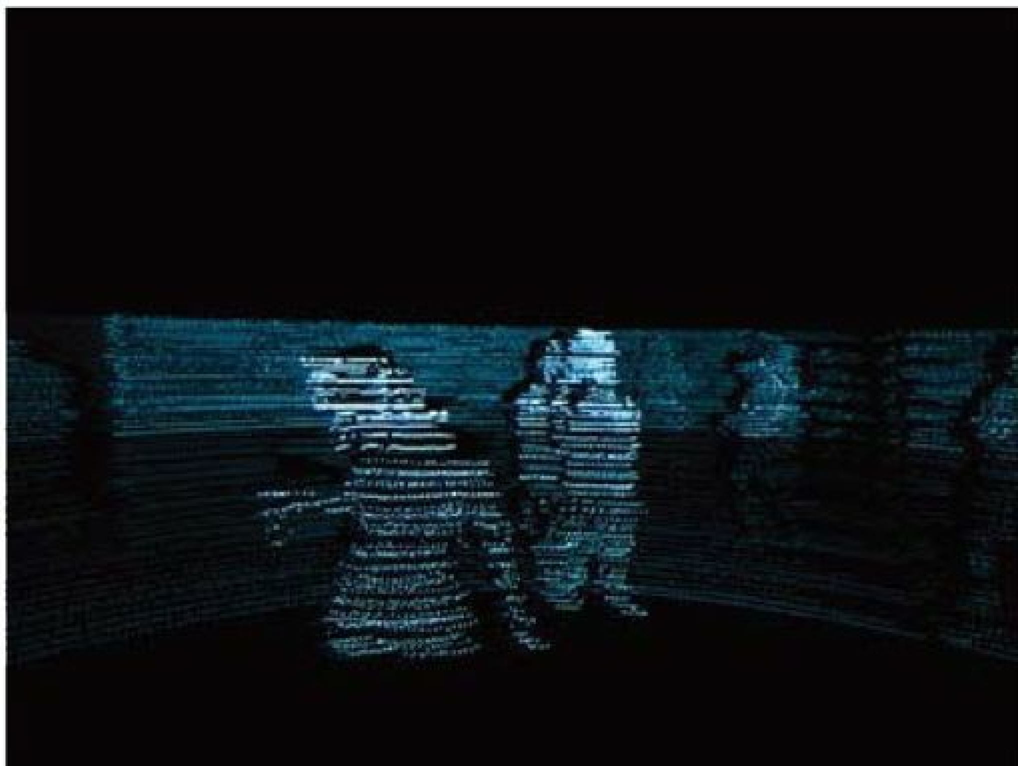
彩图　20（图6-10）　在beach基础区块中出现的对选定元素的描述符在地理标题和评论中出现的术语的空间树形图。位置向量表示图6-9中放大部分的叶子节点的绝对地理位置



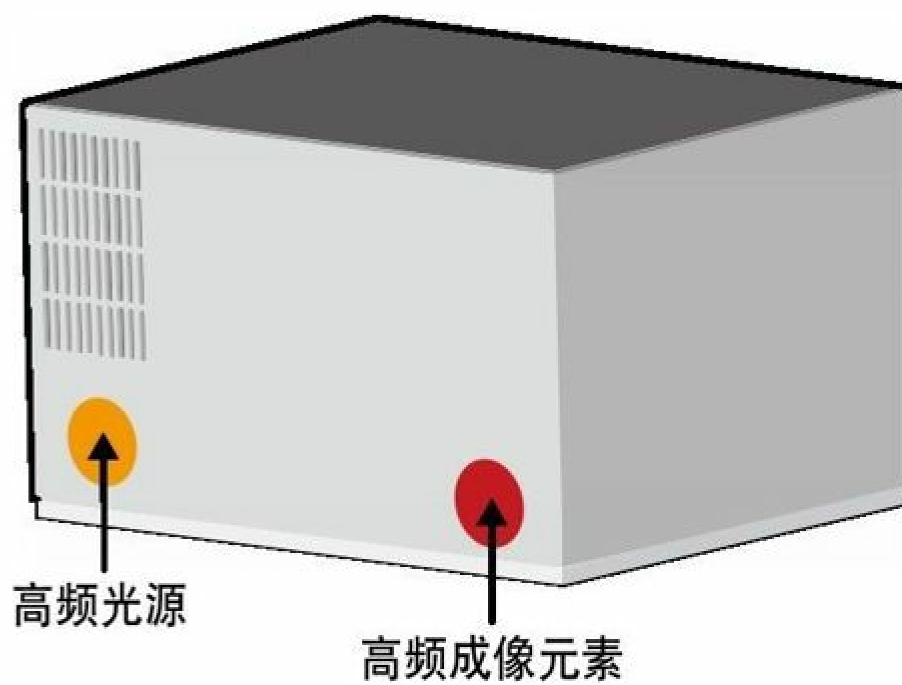
彩图　21（图9-1）　Borders商店定位表单和提交特定表单得到的Deep Web结果页



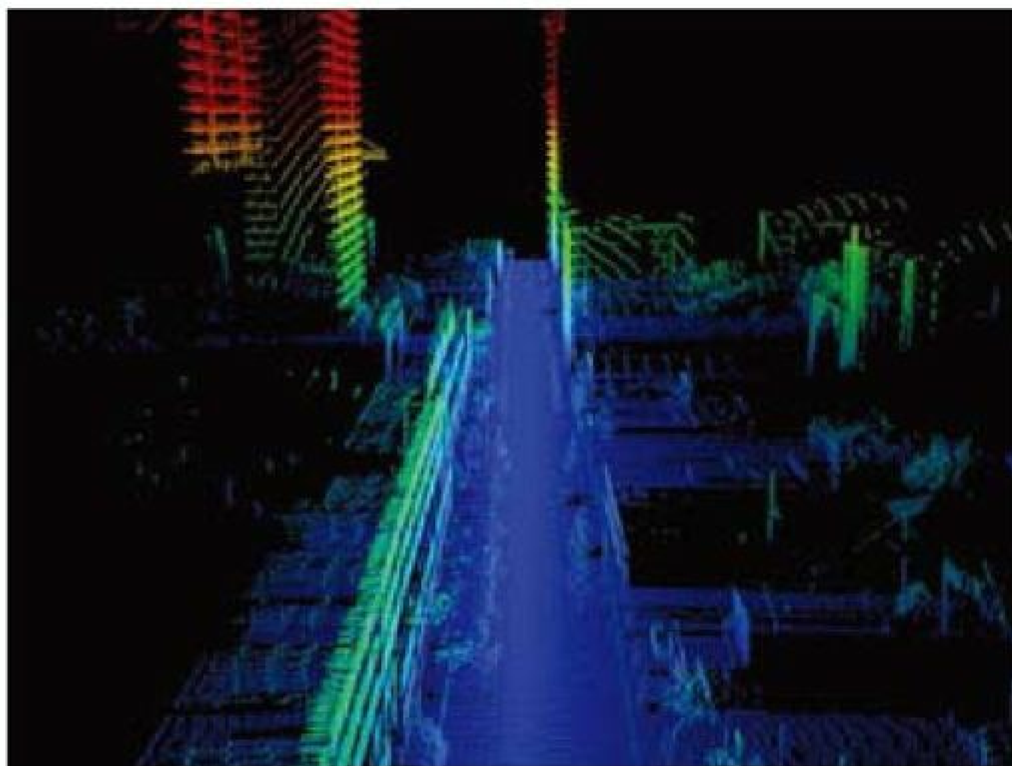
彩图　22（图10-1）　来自“飞行模式”的静态图片（2005）



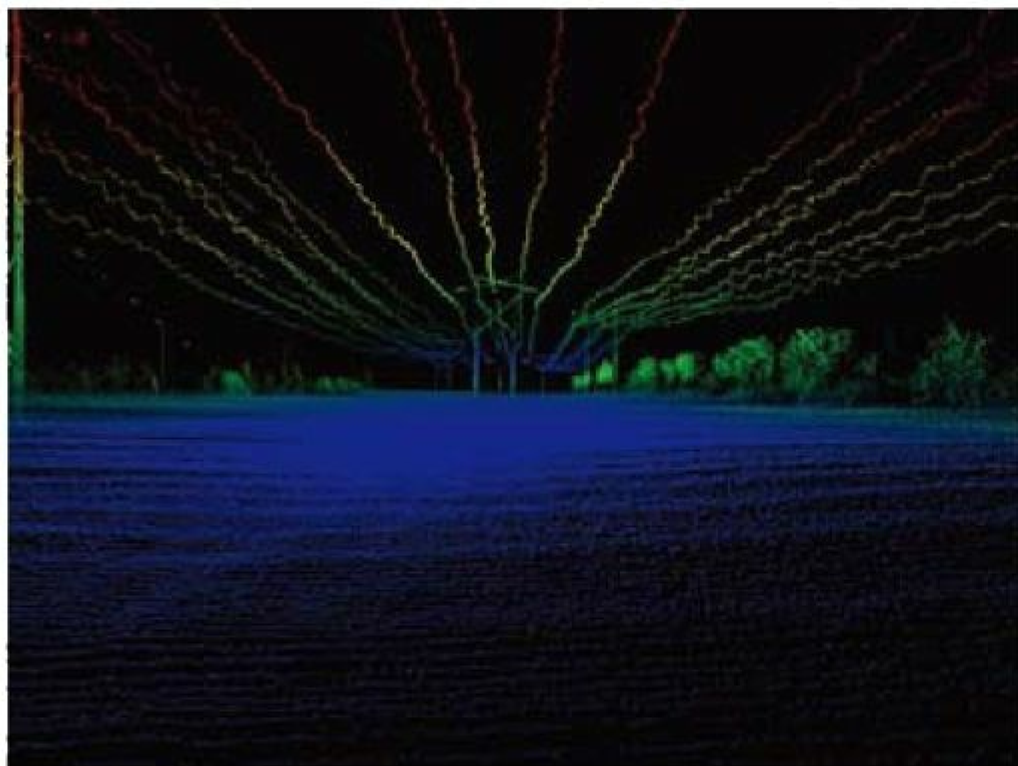
彩图　23（图10-3）　派对场景的一张静态图片，通过Velodyne的Lidar拍摄。注意该图片上方的分辨率更高，这是由于在上斜岸的激光的触发速率更快



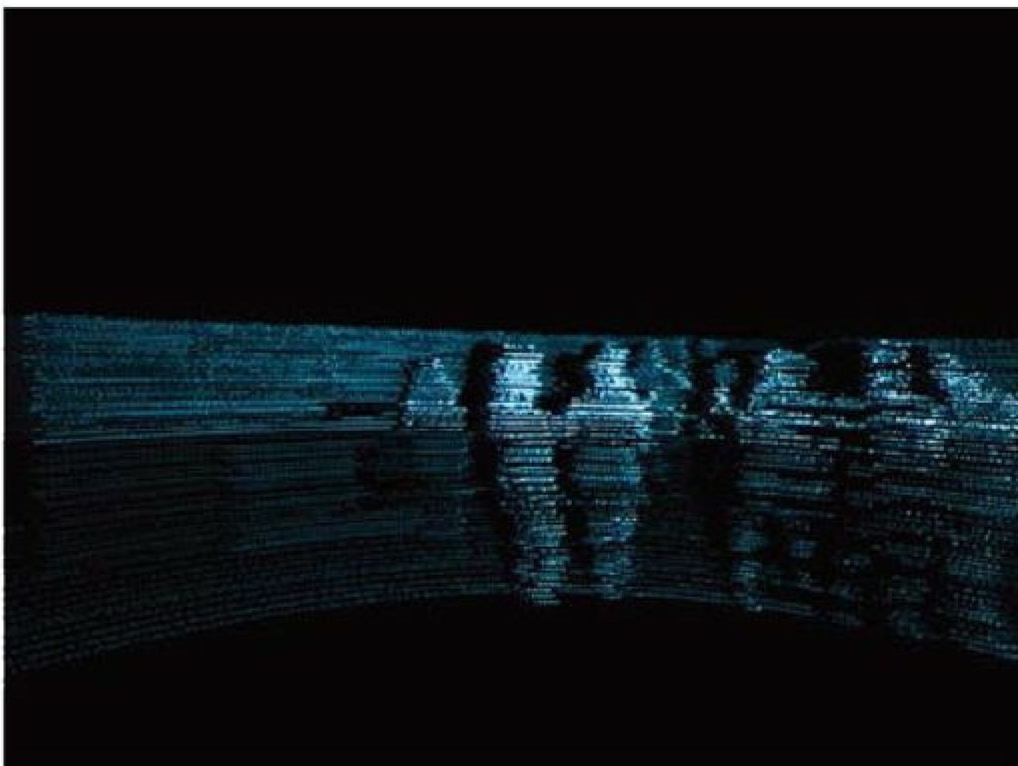
彩图　24（图10-4）　Geometric Informatics系统（图像来源：Geometric Informatics公司）



彩图　25（图10-5）　Velodyne Lidar的一个激光器捕捉到的数据



彩图　26（图10-6）　Velodyne Lidar捕捉到的另一幅景观图片



彩图　27（图10-7）　派对场景的一个静态图片，Velodyne　Lidar捕捉，数据有64条线，每条线由Lidar的一个激光器生成



彩图　28（图10-8）　歌手Thom Yorke在Radiohead视频中的一张静态图片



彩图　29（图11-2）　CrimeWatch的样本数据显示了盗窃、毒品、抢劫、盗窃车辆等犯罪活动



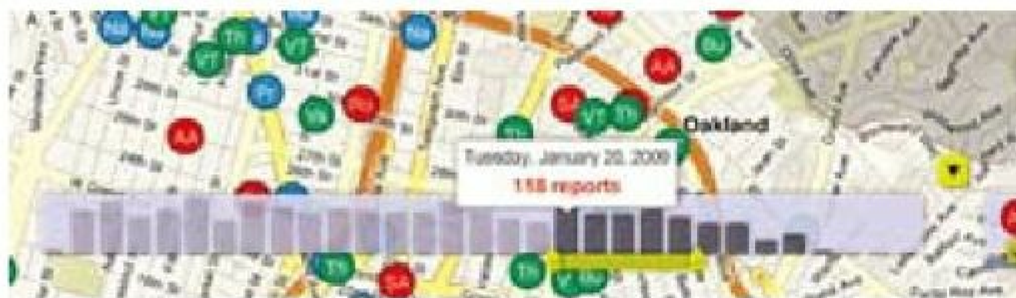
彩图　30（图11-3）　CrimeWatch相同的样本数据，突出显示了程序可识别的图标



彩图　31（图11-5）　奥克兰商业区地图，为三角测量所显示的三个参照点



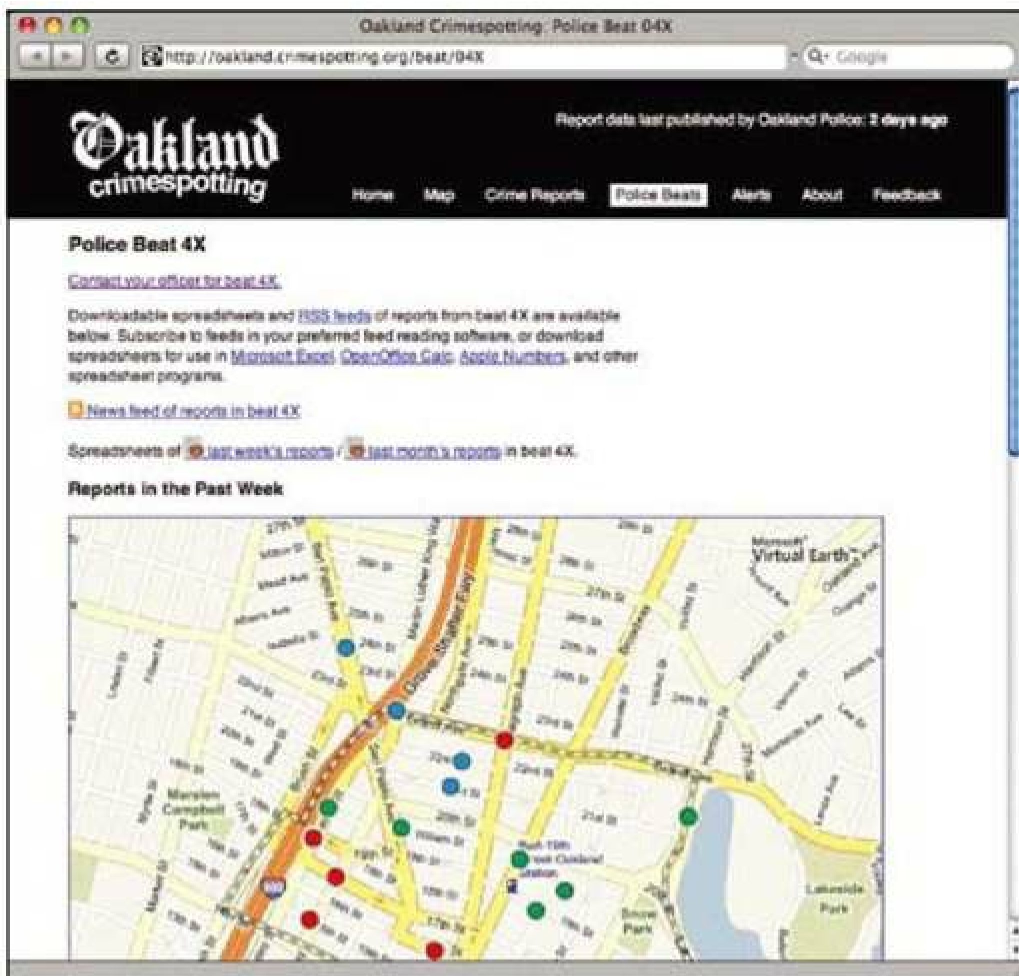
彩图　32（图11-6）　奥克兰Crimespotting项目的主页显示了从上周开始的犯罪报告地图



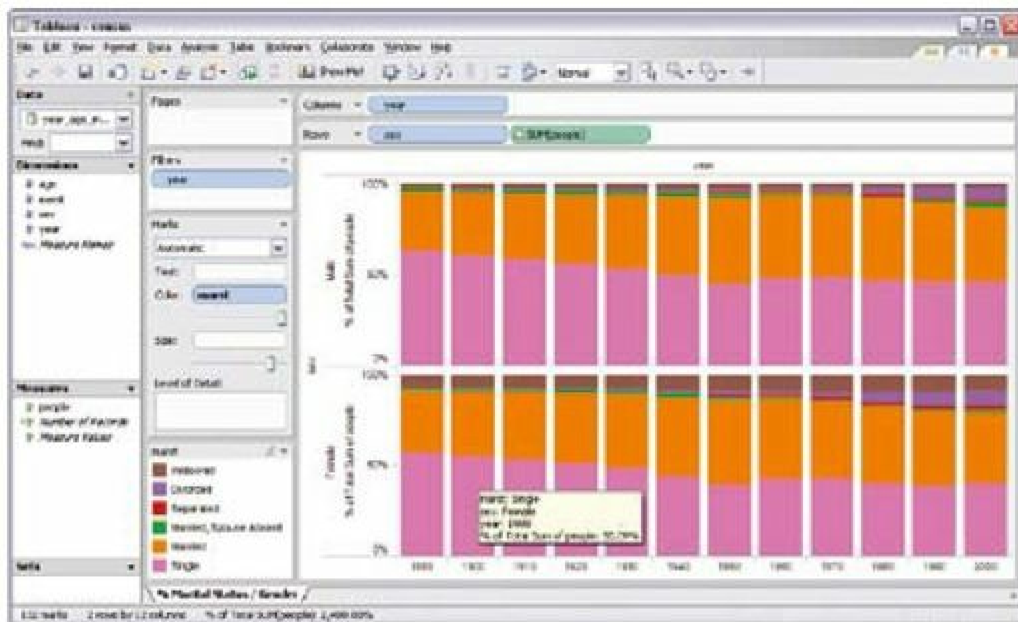
彩图　33（图11-7）　Crimespotting的主地图上的时间选择器界面



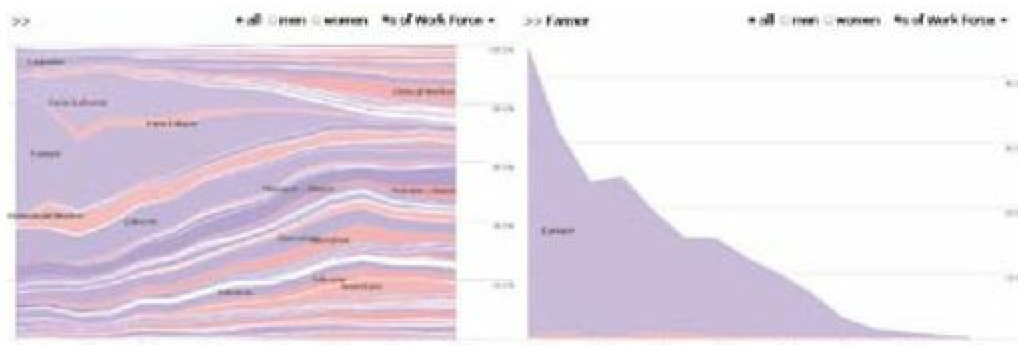
彩图　34（图11-8）　类型选择器显示了在选定时间范围内的每种报告类型的总数



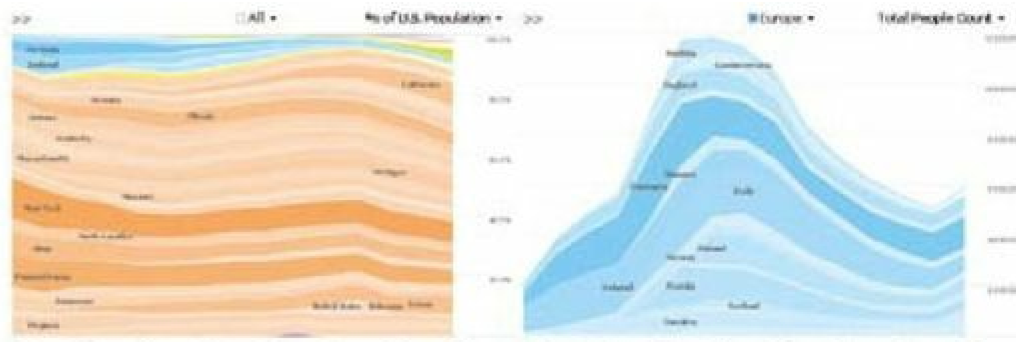
彩图　35（图11-10）　巡警区专用页面允许居民为在当地巡逻的警官提供反馈



彩图　36（图12-3）　使用Tableau工具构建的显示几十年间婚姻状态分布的原型系统可视化



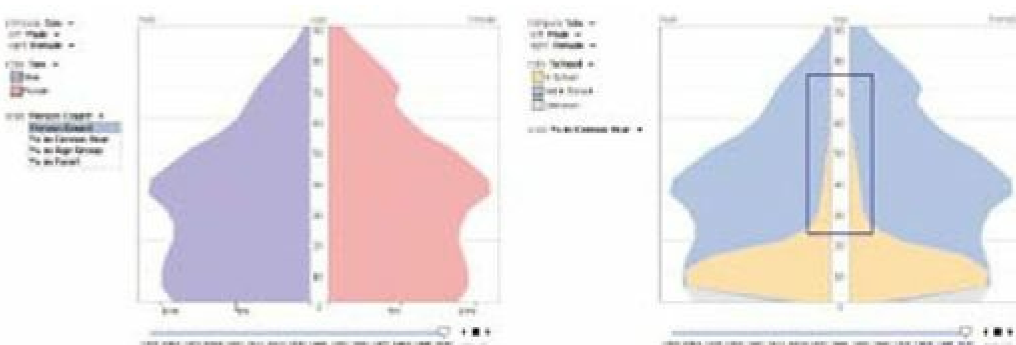
彩图　37（图12-4）　工作向导可视化：左图显示在过去150年间劳动力组成概要视图，右图显示过滤后农民工的比例的视图



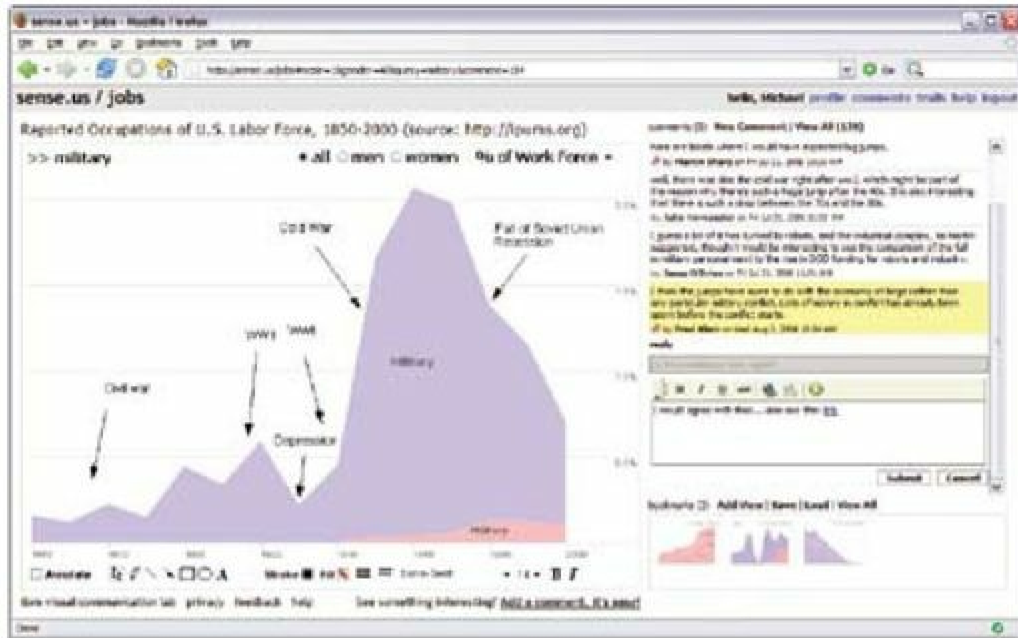
彩图　38（图12-5）　出生地向导可视化：左图显示在过去150年间出生地的分布概要，右图显示过滤后欧洲移民的总数的视图



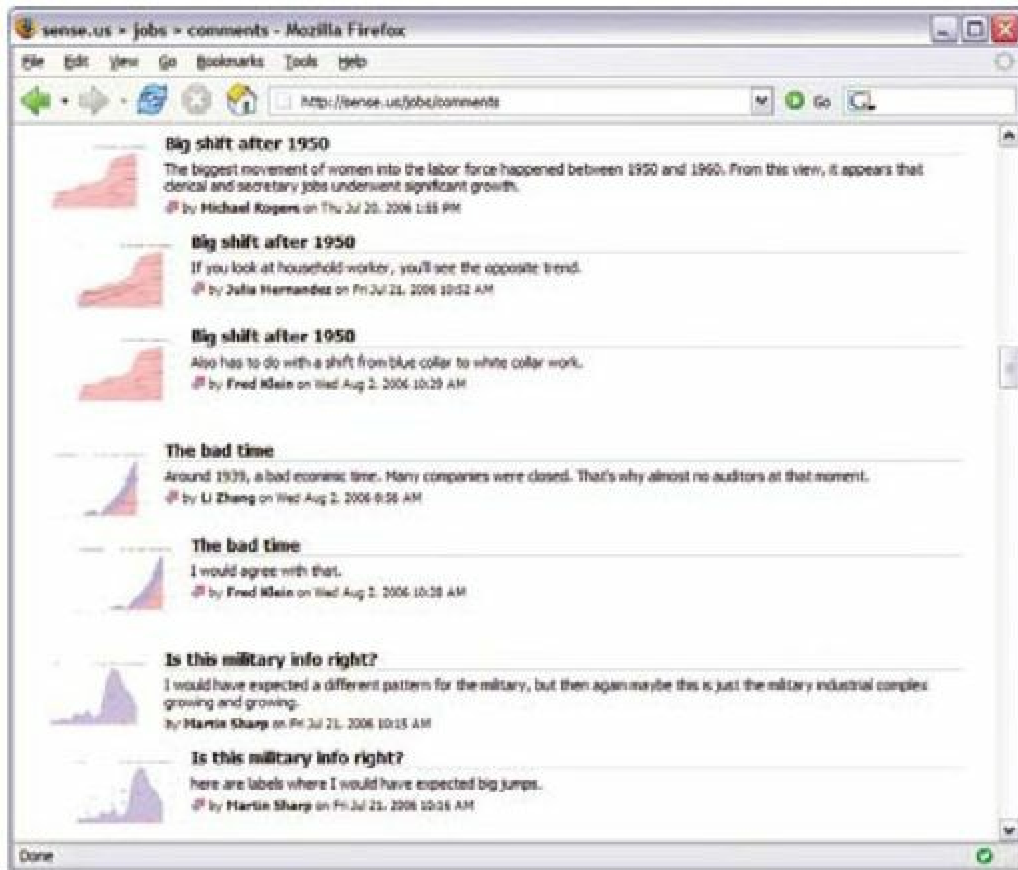
彩图　39（图12-6）　（左图）交互式国家地图显示每个州2000～2005年的人口变化，（右图）美国散点图显示了平均家庭收入（x轴）和零售额（y轴），新罕布什尔州和特拉华州有最高的零售额



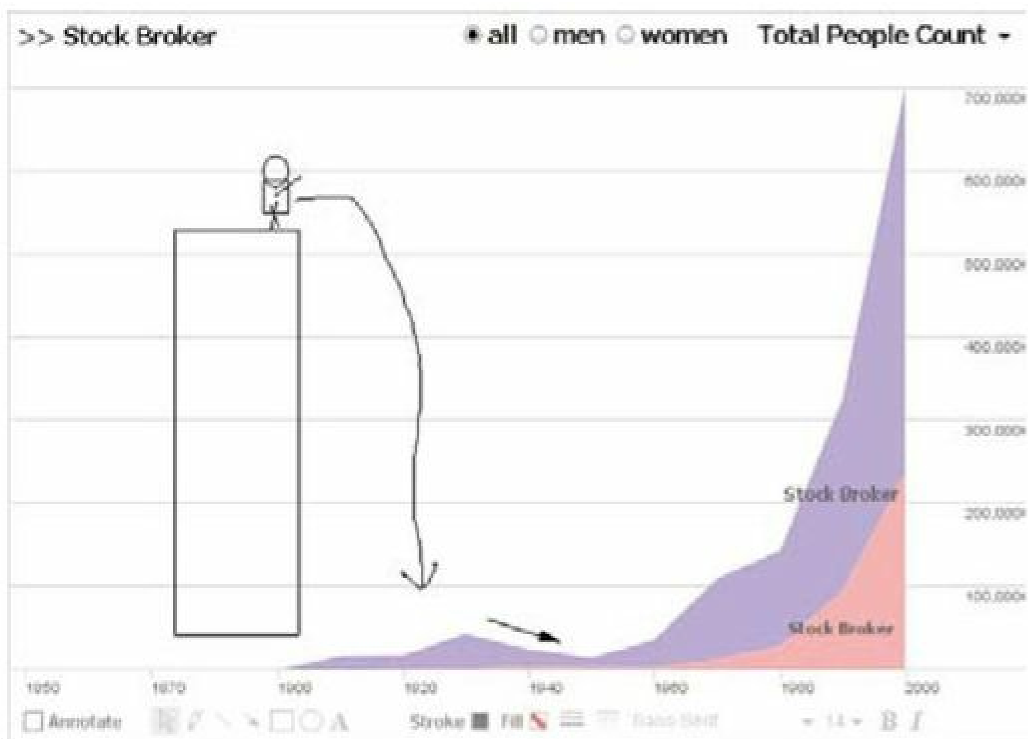
彩图　40（图12-7）　人口金字塔可视化：（左图）2000年的每个年龄组的男性和女性的总人口数量的比较，（右图）2000年学校出席人数的分布（标注重点突出了成人教育的流行率）



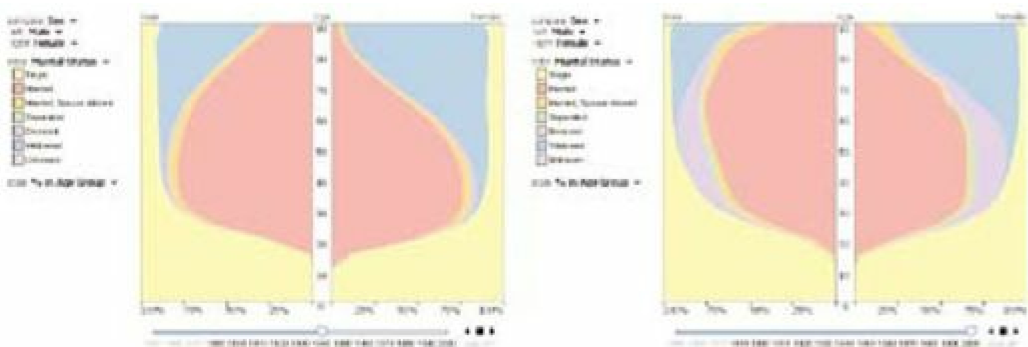
彩图　41（图12-8）　sense.us协作式可视化系统：a）交互式可视化应用小程序，图形化标注出当前选定的评论。该可视化是按照性别划分的美国劳动力人数的叠加式时间序列可视化。该图显示了军事职位的劳动力比率；b）一组图形化标注工具；c）保存的视图书签索引；d）添加评论的文本入口区域。可以拖动书签到文本区域，为评论中的视图增加链接；e）在当前视图添加相同主题的评论；f）该应用当前状态对应的URL。该URL随着可视化状态变化自动更新



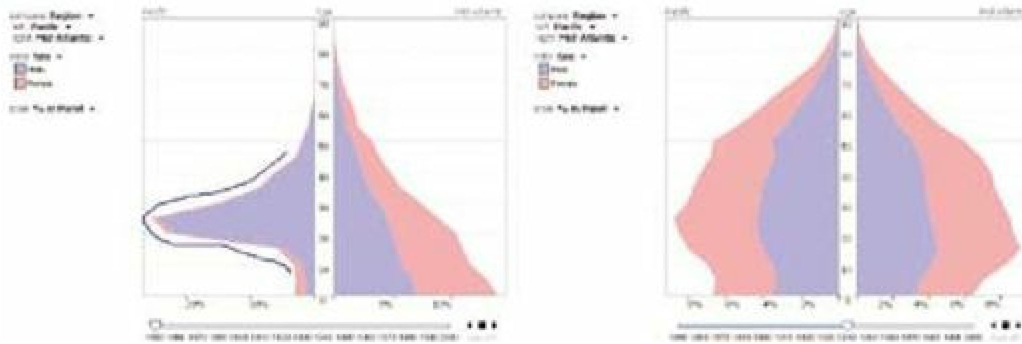
彩图 42（图12-9） sense.us评论列表页面，评论列表显示可视化的所有评论，提供链到评论的可视化视图的链接



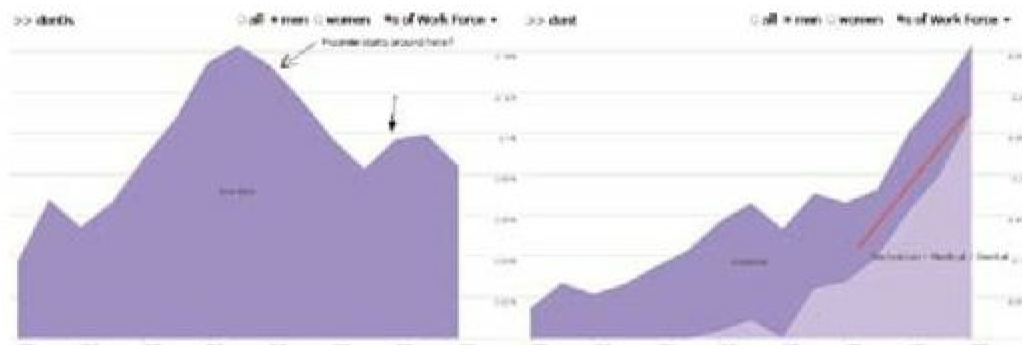
彩图　43（图12-10）　股票经纪人的标注视图；其评论意思是“大萧条‘杀死’了很多经纪人”



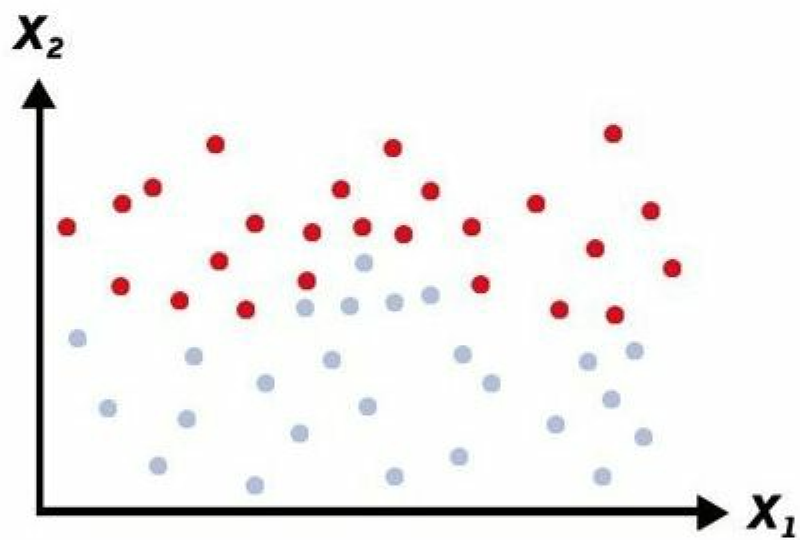
彩图　44（图12-11）　人口金字塔显示在1940年（左图）和2000年（右图）每个年龄组的婚姻状态分布



彩图　45（图12-12）　人口金字塔比较了在1850年（左图）和1940年（右图）西海岸和大西洋中部地区的人口



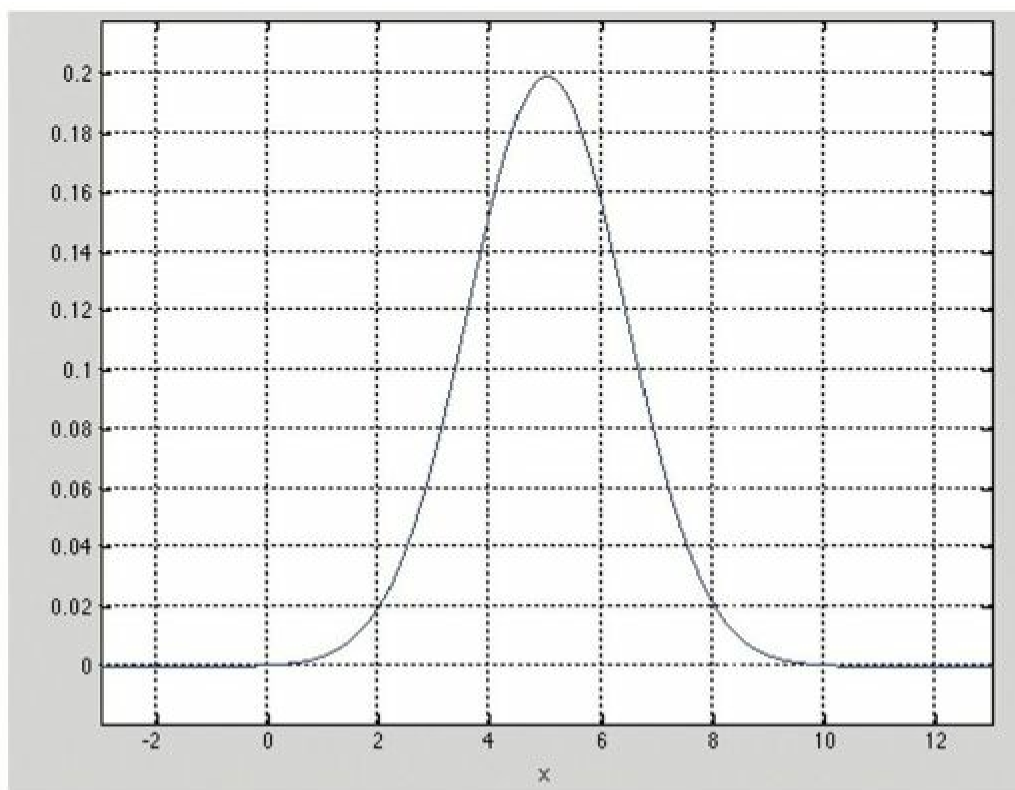
彩图　46（图12-13）　标注的工作向导视图（左图）突出显示了1930年后牙医人数的减少，（右图）由于牙医人员排序的上升，牙医从业人数得到增长



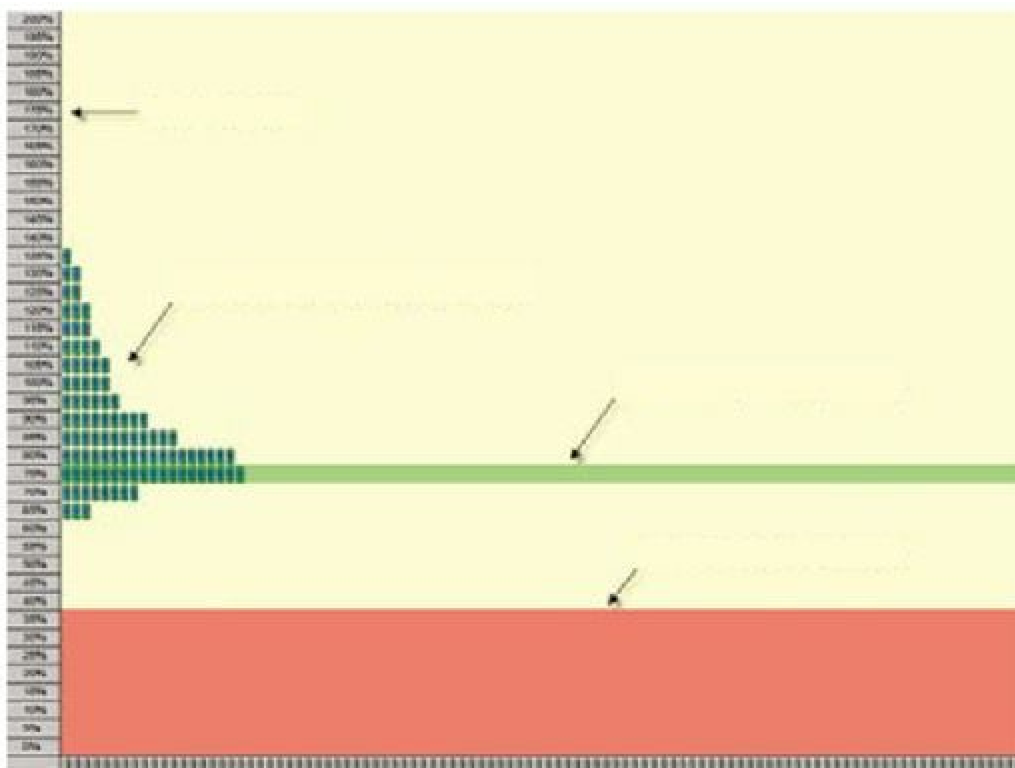
彩图　47（图13-2）　我们可以构建模型来区分两组数据集



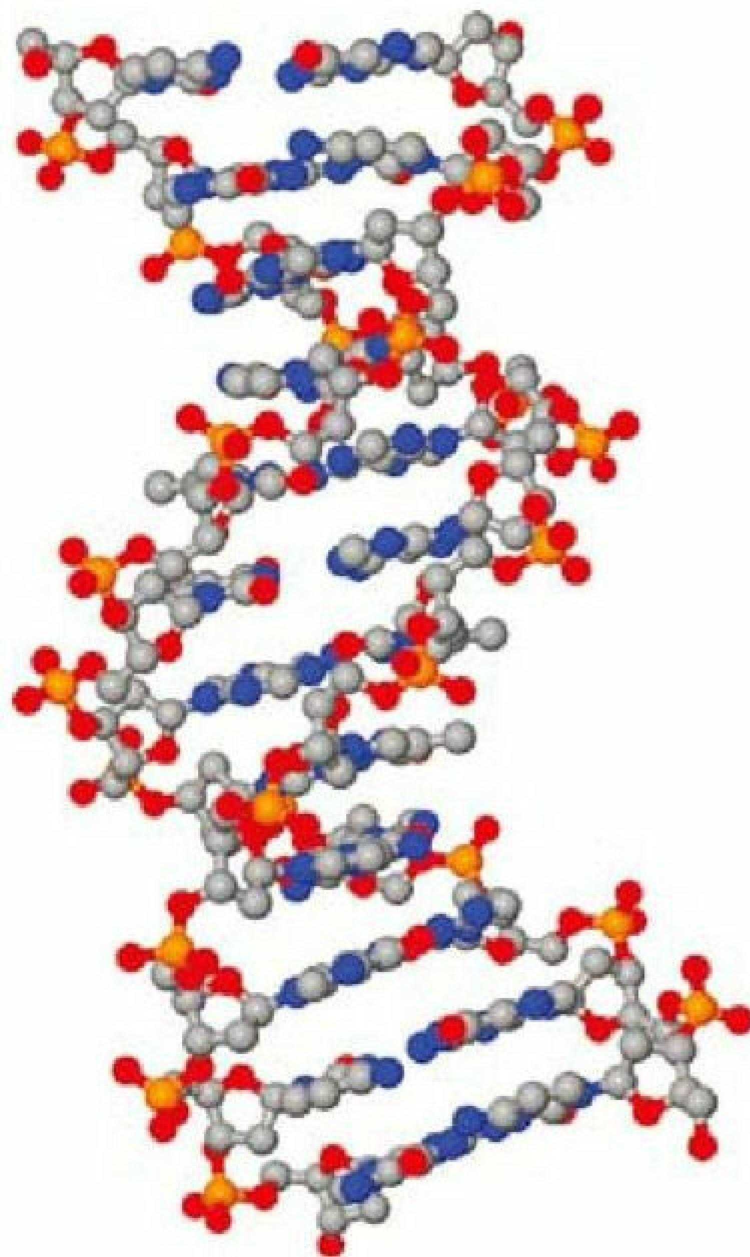
彩图　48（图13-3）　三只股票（a、b和c）在2005年的业绩



彩图　49（图13-4）　正态分布



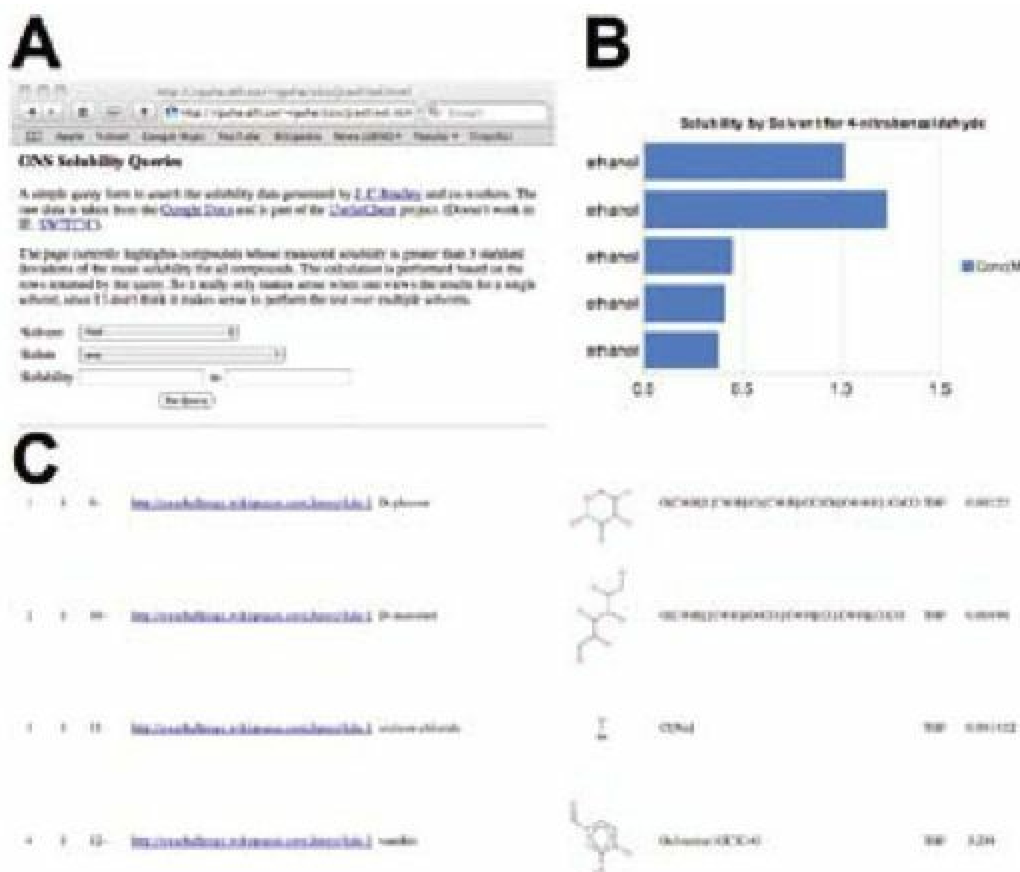
彩图　50（图13-6）　Goldstein等开发的工具帮助人们理解作为一组结果的分布



彩图　51（图15-1）　DNA中的一个小片段，由POV-Ray软件从PDB文件1BNA生成，doi：10.2210/pdb1bna/pdb

彩图　52（图15-2）　图示为一个DNA片段，其中包含一个基因，以及基因周围用来和细胞调控机制相互作用来限制基因表达的区域。这里基因由一个启动子和一个增强子引导，这样基因就被做了标记，使得细胞知道何时应当表达该基因

彩图　53（图16-1）　使用免费通用服务来托管实验工作记录及处理过的数据。A）单个实验测量时期的一部分；B）Flickr上实验中所拍照片；C）托管在Gdoc上主要数据仓库的一部分



彩图　54（图16-2）　用于检视溶解度数据的可视化工具。A）采用JavaScript和G公司Doc API生成的一个简单表单输入界面；B）溶解度数值的图形化表示；C）带有二维化学结构渲染图的数据表格输出。可以从http：//toposome.chemistry.drexel.edu/～rguha/jcsol/sol.html访问此服务。请注意：本服务与这里提到的其他服务都是动态的，可能得到与上图所示不同的查询结果

Inherent Properties, Identifiers and References

ChemSpider ID:	
Empirical Formula:	
Molecular Weight:	89.0932
Nominal Mass:	89 Da
Average Mass:	89.0932 Da
Monoisotopic Mass:	89.047678 Da

H_3C–O–C(=O)–CH_2–NH_2

Systematic Name:	methyl 2-aminoacetate
SMILES:	O=C(OC)CN
InChI:	
InChIKey:	
Std. InChI:	
Std. InChIKey:	

Associated Data Sources and Commercial Suppliers

Supplemental Information

User Data

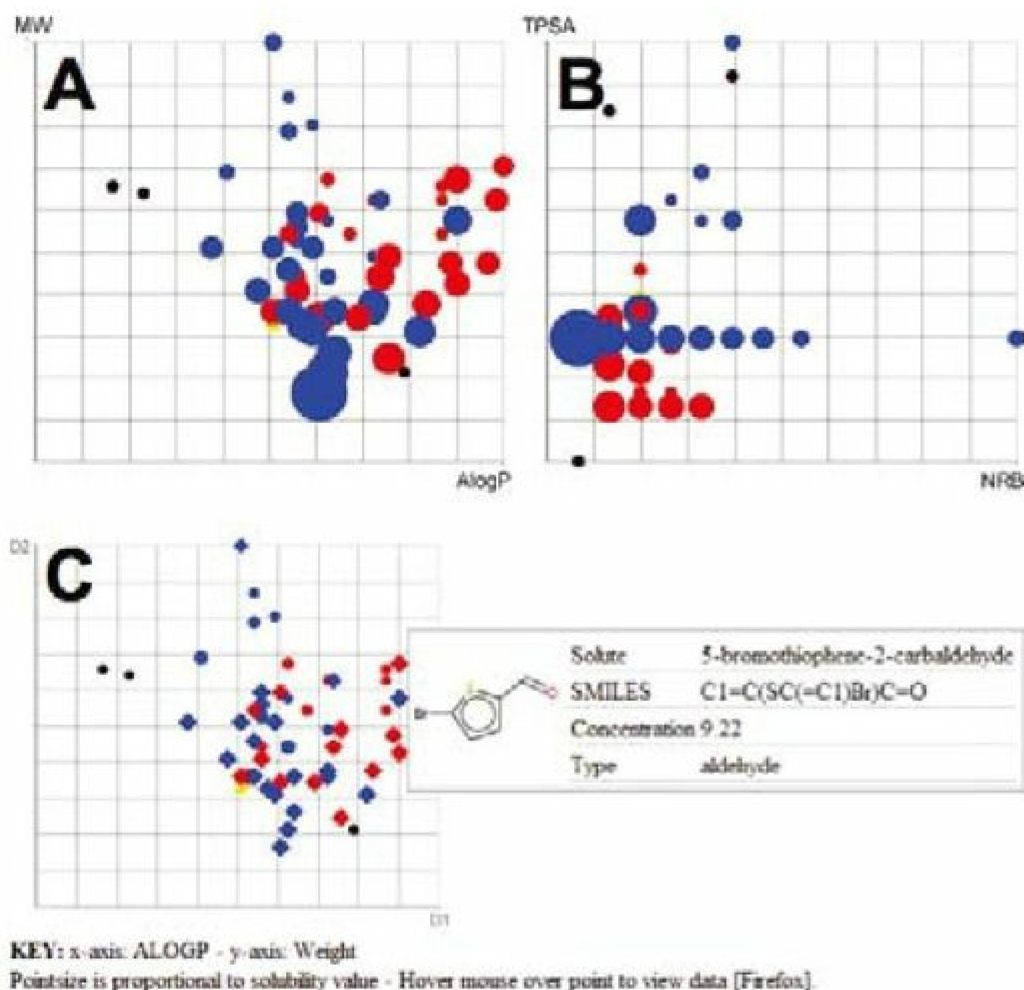
- Experimental Physchem Properties
  Non-Aqueous Solubility: 1.32M in methanol

Names and Synonyms

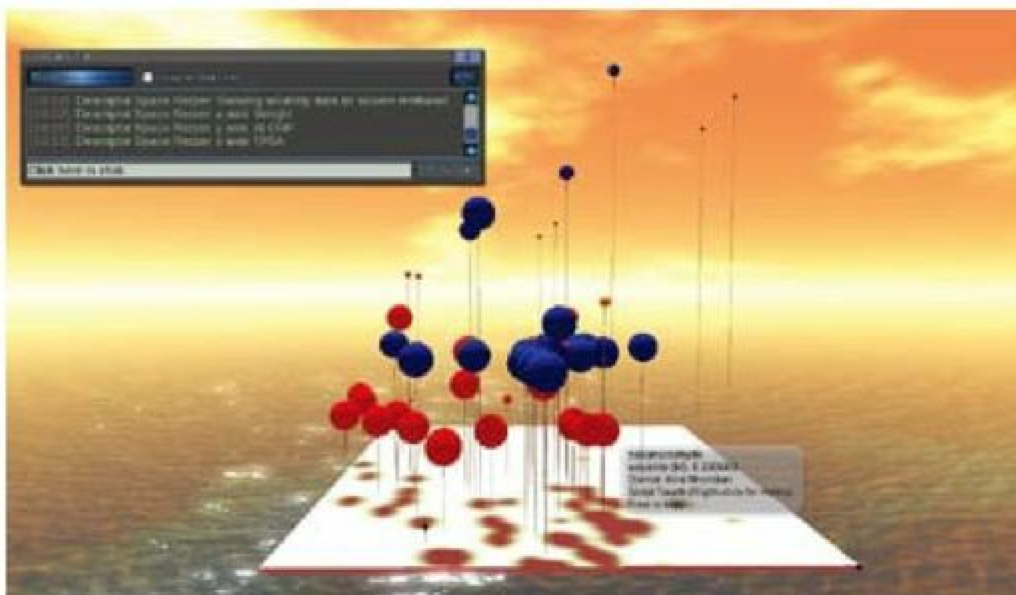
Validated by Experts, Validated by Users, Non-Validated, Removed by Users, Redirected

(Methoxycarbonyl)methylamine
glycine methyl ester
Glycine O-methyl ester
Glycine, methyl ester
Methyl aminoacetate

彩图　55（图16-3）　这是ChemSpider条目显示溶解度数据与原始数据链接的例子



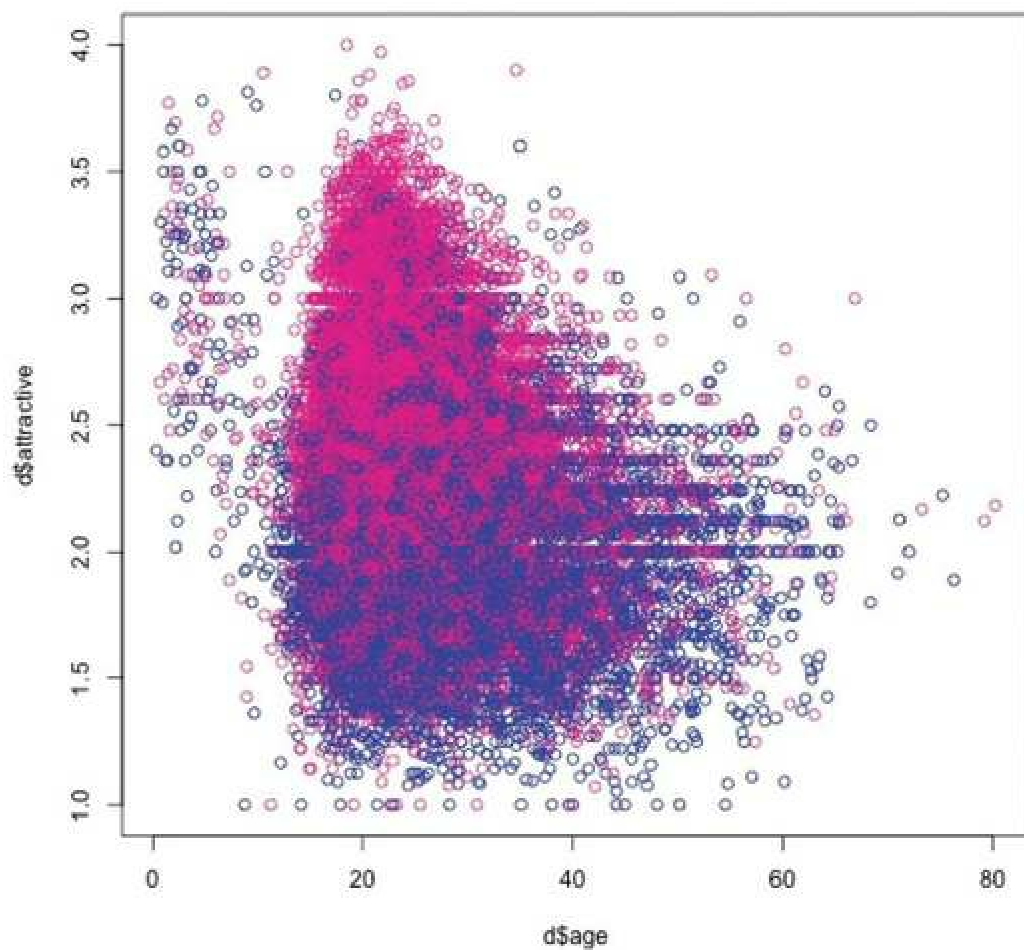
彩图　56（图16-5）　化学空间溶解度数据的图表展示。A和B给出了同一数据组在表示不同化学特性的数轴上的两个可视化图表。点的颜色表示化合物的类型（红色的为醛类，蓝色为羧酸类，黄色的为胺类，黑色表示其他），点的大小表示溶解度的值。C显示了可点击的界面，其中有单个数据点的化合物结构和溶解度值



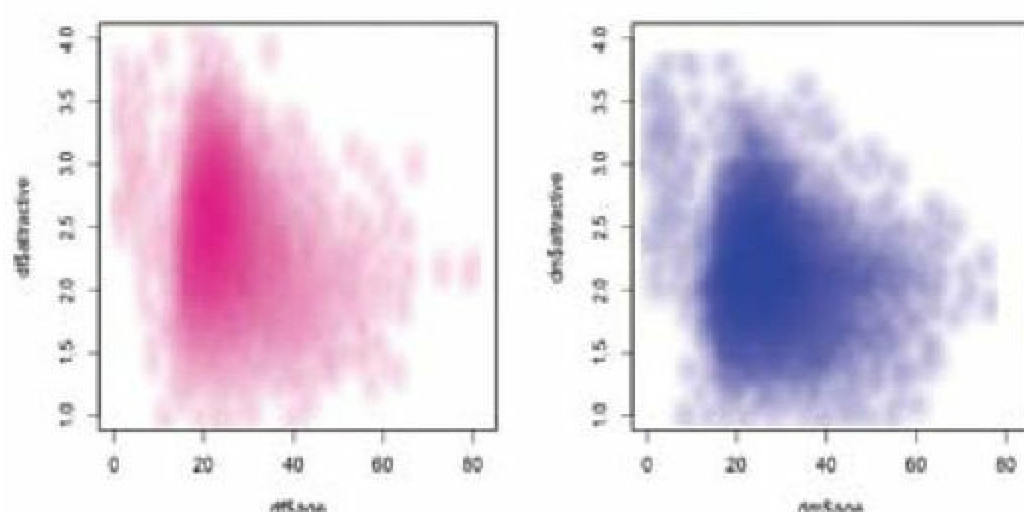
彩图　57（图16-6）　使用Second　Life展示多维数据。三个空间轴分别表示三个化学描述符。球的颜色表示化合物的类别（和图16-5的定义一样），球的大小表示在当前溶剂中的溶解度。此可视化图表可以在http：//slurl.com/secondlife/Drexel/165/178/24找到，即Second　Life中的Drexel岛



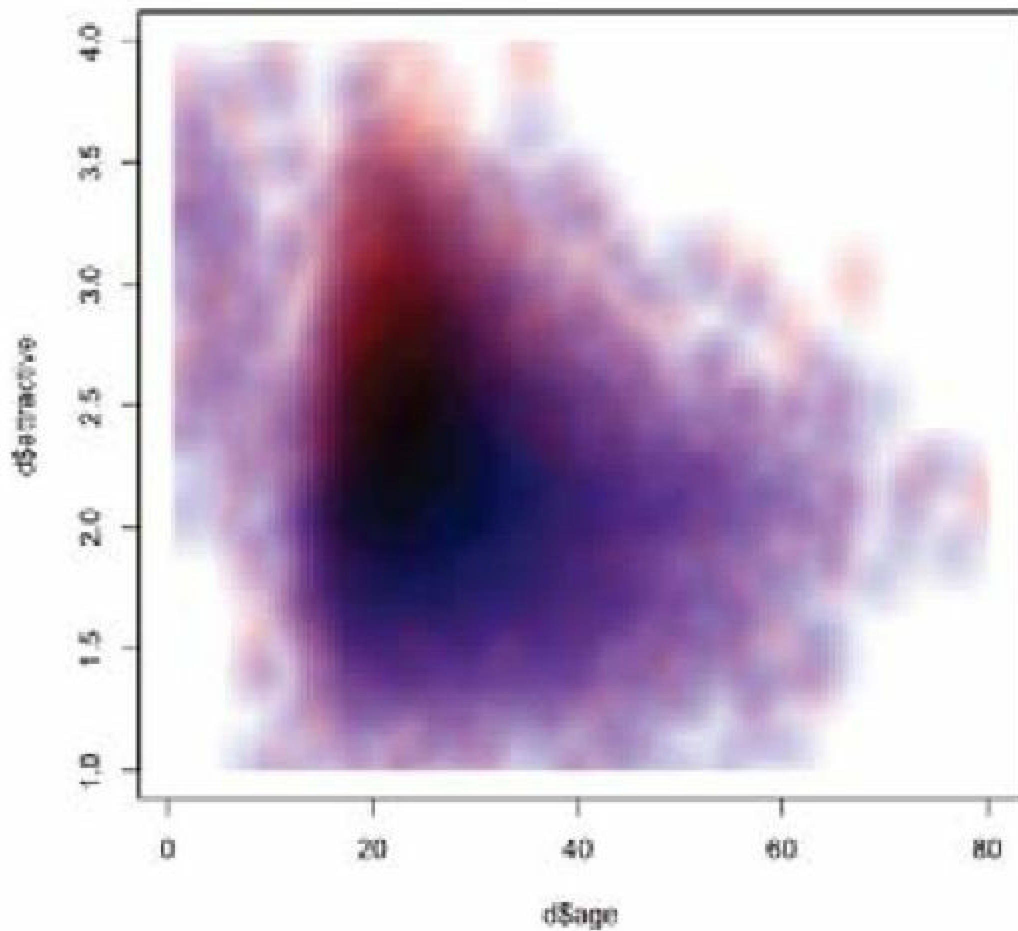
彩图　58（图17-1）　FaceStat的评判界面



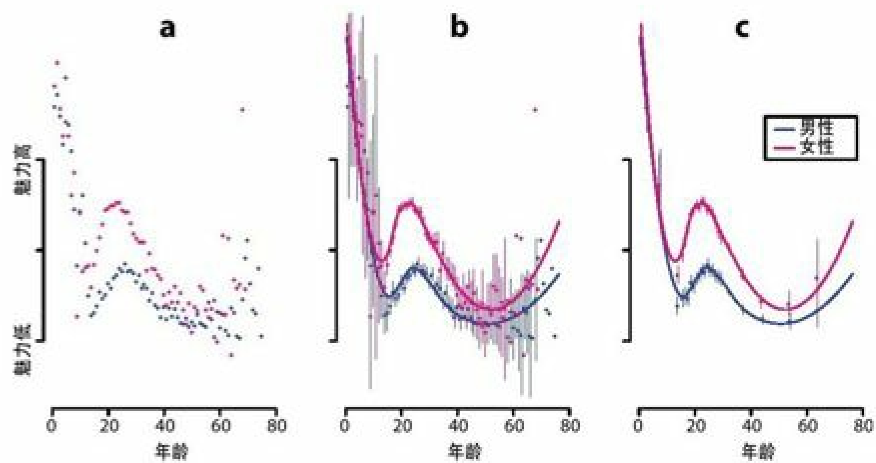
彩图　59（图17-5）　魅力和年龄关系的散点图，通过性别进行着色



彩图　60（图17-6）　魅力和年龄关系的平滑散点图，每个性别一个图

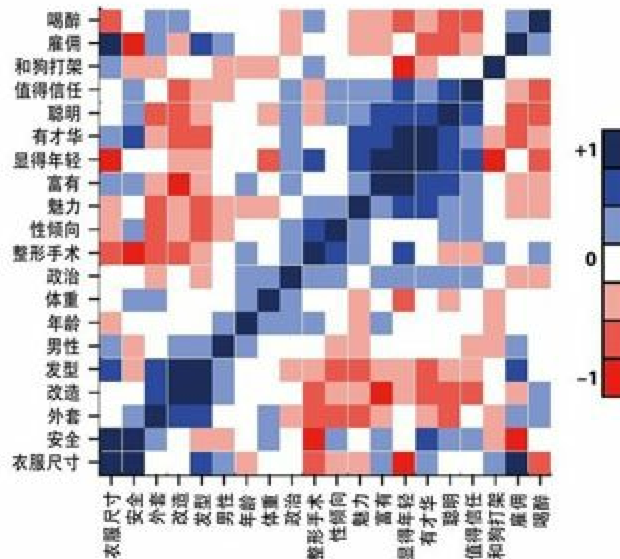


彩图　61（图17-7）　魅力和年龄关系的平滑散点图，通过性别着色，在一张图上交叠显示



彩图　62（图17-8）　魅力和年龄以及性别的关系的三种迭代绘图：

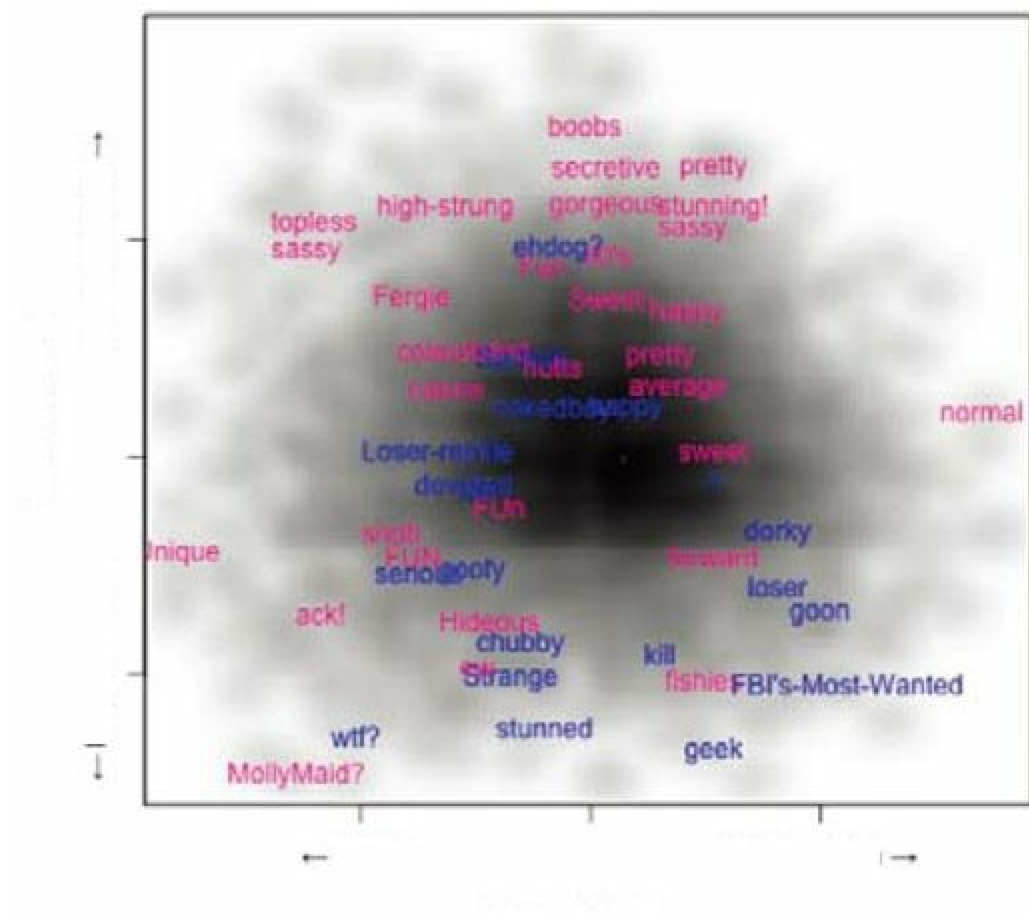
a）平均值在桶内区间的每个年龄；b）每个桶95%置信区间，以及局部加权回归曲线；c）更大的桶，其中数据更稀疏



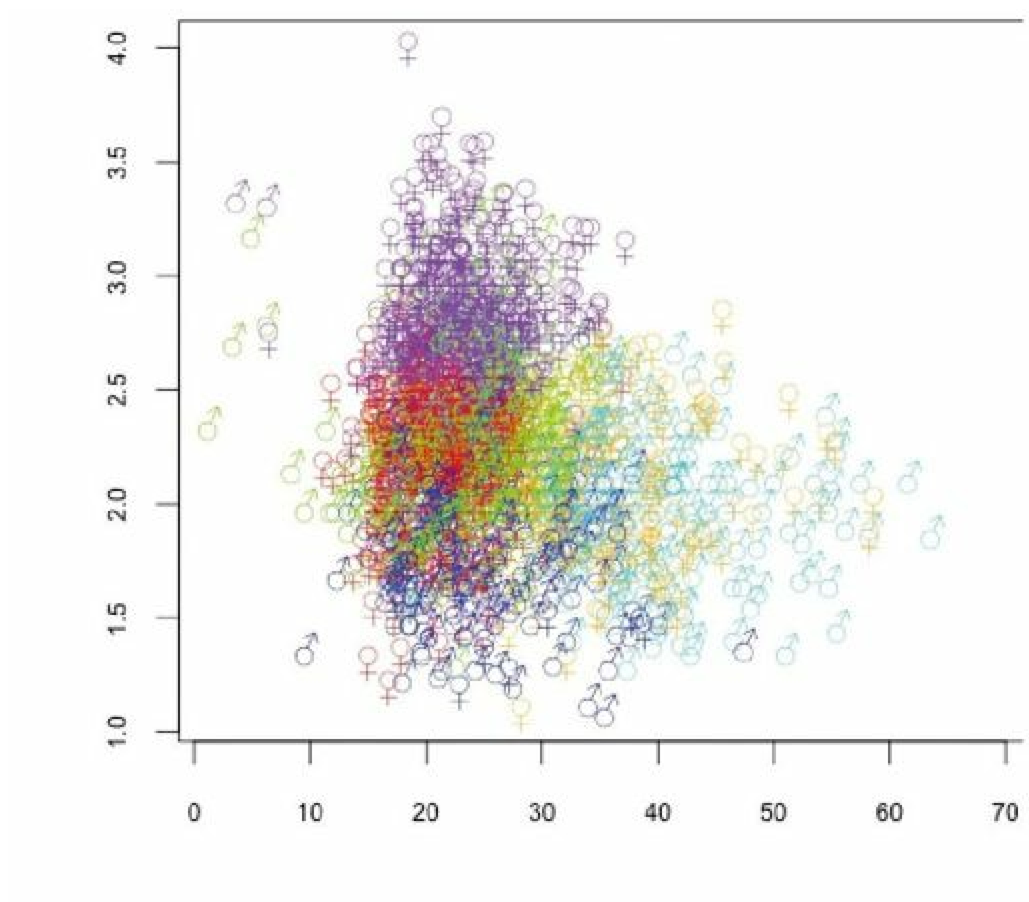
问题文本

- 衣服尺寸：我的衣服尺寸是多少？
- 安全：如果你是一名机场安检巡警，你会检查我吗？
- 外套：你喜欢我的外套吗？
- 改造：我是否需要改造？
- 发型：你喜欢我的发型吗？
- 年龄：我多大了？
- 体重：我有多重？
- 政治面貌：我的政治面貌是什么？（越高越保守）
- 整形手术：我是否做过整形手术？
- 性倾向：我的性倾向是什么？（值越高越有可能是同性恋）
- 魅力：我有多大魅力？
- 富有：我有多富有？
- 显得年轻：我是否（或者将来是否会）显得年轻？
- 有才华：我是否有才华？
- 聪明：我有多聪明？
- 值得信任：我值得信任的程度有多高？
- 和狗打架：你觉得我是否能打过一只中等大小的狗？
- 雇佣：你会雇佣我吗？
- 喝醉：我有多醉？

彩图　63（图17-9）　Pearson关联矩阵，蓝色方块的属性对和上升的斜线是正向关联，而红色方块的属性对和下降的斜线是逆相关



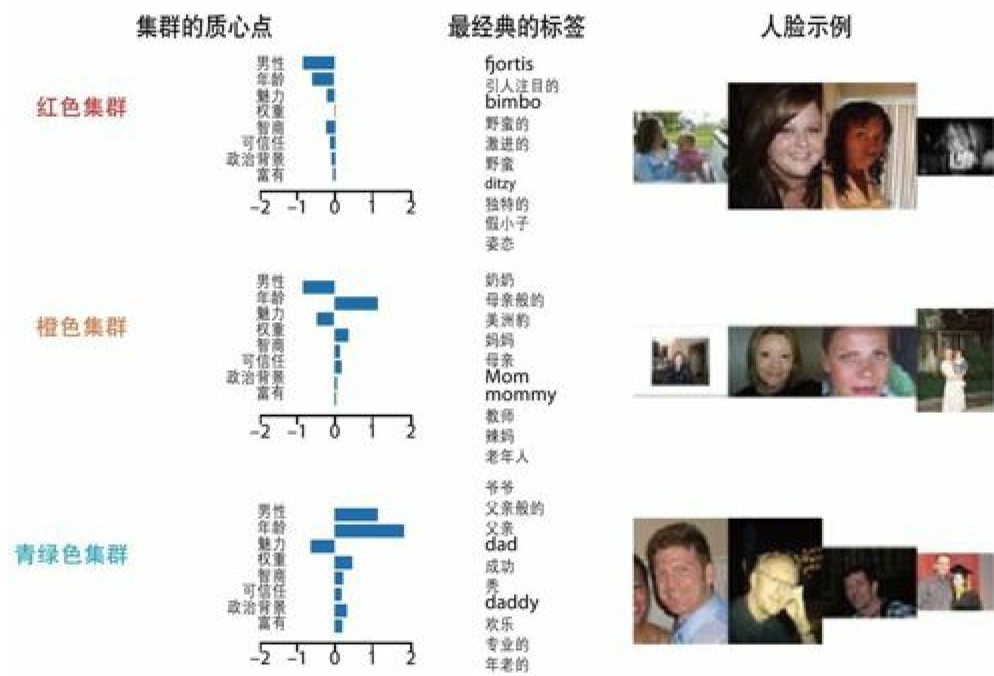
彩图　64（图17-12）　基于平滑后的魅力与年龄描绘的标签样本图



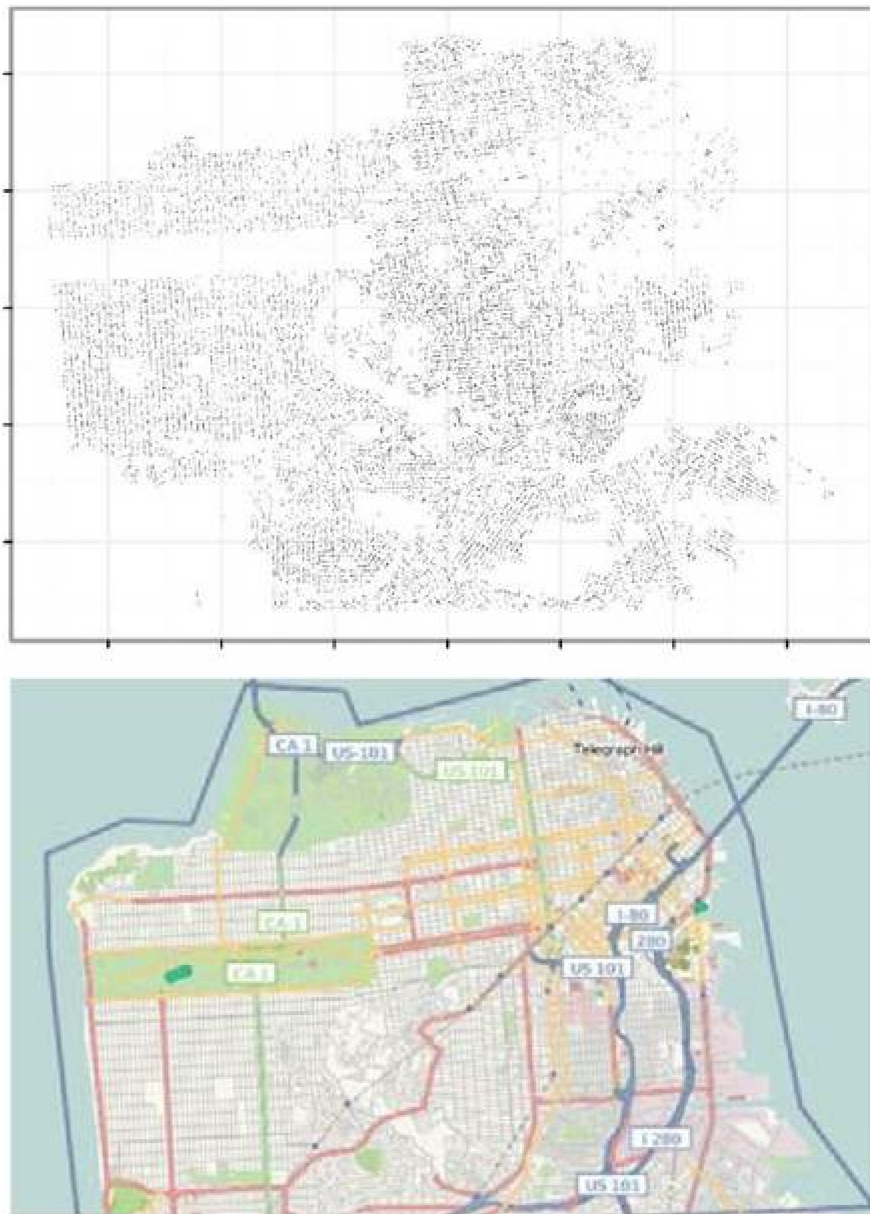
彩图　65（图17-13）　魅力和年龄的关系，通过集群着色，显示了包含2000个点的子样本



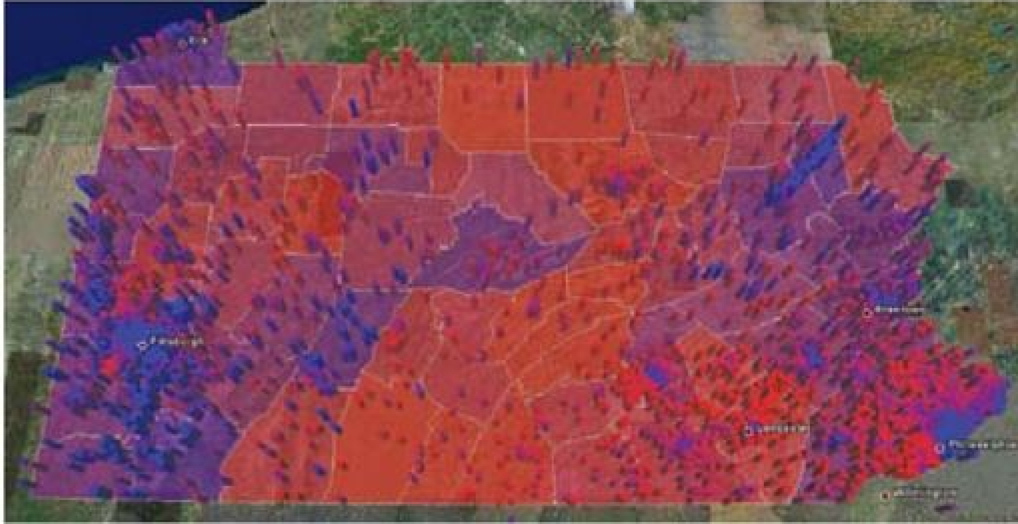
彩图 66（图17-15） 集群的质心点、标签和例子



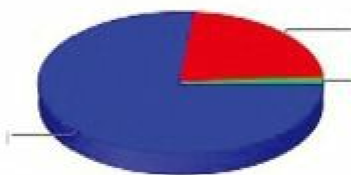
彩图 67（图17-16） 集群的质心点、标签和例子

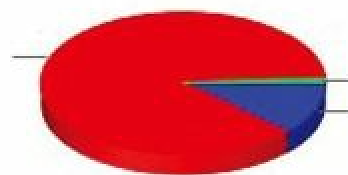


彩图　68（图18-13）　（上图）对于数据中的每个住宅销售都画出了一个小点，它给我们非常好的旧金山布局的感觉；（下图）为了比较，显示的是一张旧金山的街道地图，来自http：//openstreetmap.com



彩图　69（图19-5）　宾夕法尼亚州的地理党派。基础层显示了根据2004年总统竞选公布的数据对宾夕法尼亚州进行的分块着色显示，蓝色表示对民主党候选人John　Kerry的支持度更高，红色表示对共和党候选人George　W.Bush的支持度更高，而紫色分区表示介于这二者之间。分散的柱面图表示对于该州4000个随机注册的投票人的本地党派，定义为居住在1公里范围内支持民主党的人们的概率。每个圆柱面都是基于在投票人的住宅为中心，1公里范围内为界限，因此复制该党派衡量方式的区域。蓝色圆柱面表示更支持民主党的地区——这次考虑个人层次的注册——红色圆柱面表示更支持共和党的地区，而紫色表示介于二者之间的区域。该图之美在于它显示了在国家级别、州级别，甚至是县级别，红区和蓝区的思想观念的复杂性





彩图　70（图20-1）　证券交易委员会的数据和责任政治中心提供的政治贡献相关的数据的饼图